Codage progressif d'images par ondelettes orientées

Vivien Chappelier

N° d'ordre: 03218

THÈSE

présentée

devant l'Université de Rennes 1

pour obtenir

le grade de : Docteur de l'Université de Rennes 1
Mention Traitement du Signal

par

Vivien Chappelier

Équipe d'accueil : IRISA/TEMICS
École Doctorale : Matisse
Composante universitaire : S.P.M.

Titre de la thèse :

Codage progressif d'images par ondelettes orientées

À soutenir le 15 décembre 2005 devant la commission d'examen

M. :	Claude	Labit	Président
MM. :	Pier Luigi	Dragotti	Rapporteurs
	Marc	Antonini	
MM. :	Pascal	Frossard	Examinateurs
	Edouard	François	
	Christine	Guillemot	Directrice de thèse

En théorie, la théorie et la pratique sont identiques,
En pratique, ce n'est pas le cas.

Remerciements

Ce travail de thèse a été réalisé dans le cadre d'un contrat MENRT (Ministère de l'Éducation Nationale, de la Recherche et de la Technologie) au sein de l'IRISA (Institut de Recherche en Informatique et Systèmes Aléatoires) à Rennes.

Je tiens à remercier tout particulièrement Christine Guillemot, Directrice de Recherche à l'INRIA et responsable du projet TEMICS de m'avoir acceuillie dans ce projet et d'avoir dirigé mes travaux de thèse.

Je remercie Claude Labit, Directeur de l'IRISA, qui me fait l'honneur de présider ce jury.

Je remercie Pier Luigi Dragotti, Professeur à l'Imperial College de Londres, et Marc Antonini, Directeur de recherche CNRS, d'avoir bien voulu accepter la charge de rapporteur.

Je remercie Pascal Frossard, Professeur à l'Ecole Polytechnique Fédérale de Lausanne, et Edouard François, Ingénieur de recherche à Thomson, d'avoir bien voulu juger ce travail.

Je tiens également à remercier mes collègues de l'équipe TEMICS à l'IRISA avec qui j'ai eu le plaisir de travailler. Je remercie en particulier Teddy Furon pour ses relectures attentives de ce document, Romain Tavenard pour son travail de stage exemplaire, et Hervé Jégou pour nos discussions scientifiques fructueuses.

Je remercie aussi ma famille pour leur soutien au cours de ces trois dernières années, et particulièrement mon frère aîné pour m'avoir donné le goût des sciences et de l'informatique. Je salue par ailleurs ma filleule Blandine, ma nièce Coralie et mon neveu Loïc qui devront se contenter de cette unique page pendant quelques années encore.

J'aimerais adresser également un grand merci à mes amis, Thomas Cougnard, Cathy-France Ziouar-Cougnard, Benoît et Muriel Vaillant, Emmanuel Valiet, Fabrice Petesch et Alice Fernandez, Cédric "M. Kicourt" Courtoux, Benjamin Leblay, Coralie Loisel, Pierre Nédélec et Cécile Piedfort, Cynthia Paul, Maiwenn Flatres, Lysiane, "Mousse", Anne, "Choup"', Céline, Gwen, mes anciens collègues Jonathan Delhumeau, François Cayre, Marion Jeanne, Cécile Marc, Thomas Guillonet, et tous les autres que j'ai omis de citer ici.

Enfin j'aimerais remercier Linus Torvalds pour son système d'exploitation, Richard Stallman pour son compilateur et son éditeur de texte, Leslie Lamport pour son outil de création de documents avec lequel cette thèse a été rédigée et tous les développeurs méconnus de logiciels libres qui m'ont permis de mener à bien ce travail grâce à leurs outils de qualité.

Table des matières

Introduction

La représentation numérique des images est un problème datant des débuts de l'informatique. L'image étant un média à fort contenu sémantique ("une image, c'est mille mot"[1]), elle est devenu un moyen de communication à part entière de plus en plus présent dans notre vie quotidienne. Elle est également un outil de travail essentiel dans les domaines du bio-médical, de l'imagerie satellitaire et astronomique, de la production cinématographique, ou encore de l'informatique industrielle. L'intérêt récent du grand public pour l'image numérique, au travers des appareils photos numériques, des téléphones portables ou des ordinateurs personnels, montre que les problèmatiques liées à sa représentation, son stockage et sa transmission sont des sujets fort d'actualité.

Dans cette thèse nous nous intéresserons principalement au premier de ces problèmes, qui est celui de la représentation des images numériques. Dans la plupart des cas les problèmes liés à sa transmission ou à son stockage sont supposés indépendants et relèvent du domaine des télécommunications. La compression consiste à chercher comment décrire de manière la plus succincte possible l'information, en s'autorisant éventuellement à la dégrader. Ce traitement permet non seulement de réduire le nombre d'éléments nécessaire pour la représenter, mais également de simplifier les traitements ultérieurs en condensant l'information. Dans le cas particulier de l'image, la compression s'attache à extraire l'information visuelle des données brutes reçues de la caméra. Comme le simple fait de numériser l'image la dégrade, un autre objectif essentiel de la compression est de trouver le meilleur compromis entre la quantité d'information conservée et l'impact visuel des dégradations apportées sur l'image. Enfin, du fait de la masse des données à traiter, un compromis est souvent nécessaire entre la complexité des opérations effectuées et la qualité des résultats obtenus.

Un système de compression est constitué d'un codeur permettant d'extraire l'information de l'image pour la décrire par un nombre de taille réduite et d'un décodeur permettant de reconstituer l'image parfaitement ou approximativement à partir de l'information codée dans ce nombre. La quantité d'information à transmettre du codeur au décodeur dépend du modèle utilisé pour représenter l'ensemble des images numériques et de la connaissance a priori sur la distribution des images naturelles dans cet espace de représentation. Les codeurs actuels reposent sur des modèles bas niveau des images consistant à les représenter par une somme de nombreuses fonctions élémentaires de taille et de forme variables. Les traitements associés sont similaires à ceux effectuée par

[1] proverbe attribué à Napoléon

notre cortex visuel primaire pour analyser les images que nous percevons tous les jours. Nous sommes cependant bien loin de savoir exploiter de manière informatique la masse de connaissances de plus haut niveau que notre cerveau utilise pour appréhender les images naturelles.

Les systèmes de compression d'image les plus performants à l'heure actuelle reposent sur trois étapes de traitement (Fig. 1). Pour les images couleur, une étape préliminaire supplémentaire consiste à séparer l'information de couleur de l'information d'intensité lumineuse. Bien que la problématique de représentation efficace des couleurs soit un domaine d'étude pertinent, nous ne nous intéresserons pas à cet aspect là dans cette thèse, et ne considérerons donc que des images en niveaux de gris. La première étape de codage consiste alors à changer d'espace de représentation de l'image. En effet, l'intensité lumineuse des pixels constituant les images naturelles varie généralement très peu localement et n'en constitue pas une représentation compacte. Ainsi, l'étape de *transformée* consiste à appliquer une transformation généralement linéaire aux valeurs d'intensité représentant l'image afin d'en extraire les composantes principales. Parmi les approches les plus performantes dans ce domaine, les décompositions en ondelettes consistent à représenter l'image par une somme de petites vagues d'intensité, de taille et de position variables. Ces approches représentent toutefois assez mal les contours de l'image ce à quoi nous tenterons de palier dans cette thèse en proposant d'autres transformées.

La deuxième étape consiste à trouver un représentant adéquat de l'image observée parmi un dictionnaire de représentants autorisés. Cette étape de *quantification* dégrade généralement le signal car tout représentant ne peut être autorisé si l'on veut pouvoir représenter l'information par un nombre de taille finie. Les techniques de quantification utilisées dans les systèmes de compression d'image actuels sont généralement assez simples. Les plus performantes reposent sur des techniques issues du domaine des télécommunications, en particulier du codage correcteur d'erreurs. Nous présenterons dans cette thèse une extension de ces dernières techniques de quantification à des codes plus performants appelés codes turbo.

Enfin, la dernière étape permet de construire le nombre représentant l'image à partir de la connaissance a priori sur la probabilité d'obtenir chaque représentant du dictionnaire de quantification. Les codeurs de sous-bande utilisés en compression d'image reposent sur un codage progressif de ce nombre et un apprentissage de sa probabilité d'apparition au cours du codage. Nous utiliserons par la suite les codeurs existants en les adaptant aux modifications apportées en début de chaîne. Ceci nous permettra d'évaluer les performances finales des approches proposées dans un cadre concret d'application.

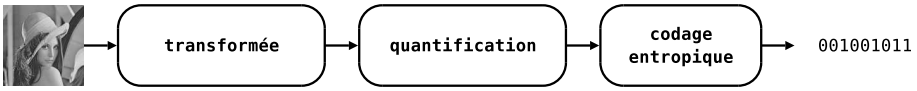

FIG. 1: Système de compression par transformée.

Le succès des transformées en ondelettes en compression d'image au début des années

90 est dû principalement à deux raisons. D'une part l'émergence de codeurs de sous-bandes capables de représenter efficacement et simplement les coefficients d'ondelettes a permis d'obtenir des performances (en termes de qualité d'image par rapport au débit) bien meilleures que celles des codeurs existants à cette époque. D'autre part, la transformée en ondelettes, couplée aux codeurs de sous-bandes correspondants, offre une représentation naturellement progressive de l'image. Ceci signifie qu'il est possible d'obtenir une image de moindre qualité ou de plus faible résolution en tronquant le flux binaire obtenu en sortie du codeur, et ce sans perte de performance. Ainsi, il est inutile de coder l'image plusieurs fois en fonction des résolutions ou des débits désirés. Cette approche apporte de nombreuses fonctionnalités. Par exemple, une image codée progressivement peut être décodée sans complexité supplémentaire à la résolution du terminal sur lequel elle est visionnée. Lors d'une transmission sur un réseau de capacité variable, il est également possible de tronquer le flux binaire représentant l'image pour l'adapter au débit disponible, sans nécessiter un décodage suivi d'un ré-encodage. Enfin, il est possible d'afficher une version approximative de l'image sans attendre de l'avoir décodée entièrement. Ces codeurs de sous-bandes adaptés à la transformée en ondelettes ont donc apporté à la fois un gain en qualité et en termes de possibilités d'utilisation.

Ainsi, seule la chaîne de codage considérée dans son ensemble permet d'évaluer les performances de ces systèmes de compression d'images. Dans cette thèse nous nous intéresserons donc aux différentes étapes la constituant. Nous commencerons tout d'abord au chapitre 1 par établir un cadre théorique commun aux techniques étudiées. Nous poursuivrons en présentant les techniques actuelles de transformées et de codage de sous-bandes dans le chapitre 2. Nos contributions seront exposées au travers des trois chapitres suivants. Nous proposerons tout d'abord au chapitre 3 une adaptation et une amélioration de la transformée en contourlettes dans le cadre de la compression d'images et de vidéos. Après avoir présenté cette transformée et les filtres la constituant, nous la combinerons avec une transformée en ondelettes séparables pour en réduire la redondance. Une technique de projection sur ensembles convexes sera ensuite utilisée afin de l'optimiser pour la compression. Nous présenterons ensuite au chapitre 4 une nouvelle transformée en ondelettes basée sur une carte d'orientation et une structure d'échantillonnage quinconce. Nous appliquerons cette transformée non redondante à la compression et au débruitage d'image, en optant pour une structure de carte d'orientation adaptée à l'application. Nous étudierons également la dépendance résiduelle entre les coefficients dans le but de concevoir des codeurs entropiques efficaces. Nous terminerons au chapitre 5 en présentant une nouvelle technique de quantification basée sur des codes turbo. La quantification codée par treillis sera introduite dans ce chapitre, puis adaptée aux codes turbo. Nous y étudierons également la convergence de l'algorithme itératif d'estimation dans le cadre de la quantification. L'intérêt d'un tel quantificateur pour d'autres applications que la compression sera aussi évoqué.

Notations

Dans l'ensemble de cette thèse, nous utiliserons les notations suivantes pour représenter les divers objets mathématiques que nous aurons à manipuler.

Afin de simplifier les développements analytiques nous nous placerons dans le cadre général des distributions et de la théorie de la mesure. Les objets de nature continue ou discrète seront donc traités avec les mêmes outils mathématiques ; en particulier les intégrales seront prises au sens de Lebesgue.

Les variables aléatoires seront notées en majuscule, tandis que leur réalisation sera notée en minuscule. Pour simplifier les notations, la probabilité $\mathbb{P}(X = x)$ que la variable aléatoire X soit égale à x sera notée $\mathbb{P}(X)$. L'espérance d'une variable aléatoire X sera notée $\mathbb{E}[X]$.

Les vecteurs et les matrices seront notées en caractères sans empatement, comme par exemple un vecteur v et une matrice A. Les composantes des vecteurs et des matrices seront notées en les indexant dans l'ordre ligne colonne. Ainsi les deux notations suivantes $\mathsf{A}_{i,j} = \mathsf{A}[i][j]$ représentent la valeur du scalaire figurant à la $(i+1)$-ème ligne et $(j+1)$-ème colonne de la matrice A. Les vecteurs sont vus comme des matrices à une seule colonne et d lignes où d est la dimension du vecteur considéré. L'indice de colonne sera omis pour les vecteurs.

Les diverses notations suivantes seront également utilisées par la suite :

- $[\![a, b]\!]$: ensemble des entiers entre a et b inclus.
- $|\mathbb{A}|$: cardinal de l'ensemble $\mathbb{A}$.
- $\mathcal{M}_{k,l}(\mathbb{A})$: ensemble des matrices k lignes l colonnes à éléments dans $\mathbb{A}$.
- $\mathcal{M}_k(\mathbb{A})$: ensemble des matrices carrées de taille k à éléments dans $\mathbb{A}$.
- $\mathbf{GL}_k(\mathbb{A})$: ensemble des matrices carrées de taille k inversibles à éléments dans $\mathbb{A}$.
- $\mathsf{M}^\top$: transposée de la matrice M.
- $\mathsf{M}^{-\top}$: inverse de la transposée de la matrice M.
- $f[n]$: valeur au point n du signal discret f défini sur $\mathbb{Z}$.
- $f(t)$: valeur au point t du signal continu f défini sur $\mathbb{R}$.

Chapitre 1

Cadre théorique

Les techniques présentées dans cette thèse se placent à différents niveaux de la chaîne de compression. Cette chaîne est constituée de trois étapes principales que sont la transformée, la quantification et le codage entropique. La transformée a pour but de décorréler le signal à coder. Ceci simplifie les traitements en aval en diminuant la dépendance statistique du signal traité. L'étape de quantification consiste à trouver parmi un ensemble fini de représentants, celui qui minimise la distorsion introduite sur le signal à valeurs réelles à quantifier. Cette étape repose sur la théorie débit-distorsion qui définit le lien entre la distorsion introduite et le coût de codage du représentant. Enfin la dernière étape consiste à coder (sans perte) la valeur du représentant choisi en un minimum de bits.

Les transformées proposées dans les chapitres 3 et 4 se placent dans le cadre général de l'analyse par ondelettes discrètes en dimension finie. Les techniques de quantification présentées au chapitre 5, quant à elles, reposent sur la théorie débit-distorsion et la théorie des lattices. Ce chapitre a donc pour but de présenter de manière synthétique les outils théoriques ainsi que les résultats fondamentaux qui seront utilisés par la suite.

Nous présentons tout d'abord le cadre théorique des lattices qui permet également d'aborder l'échantillonnage en dimension finie de manière élégante. A l'aide de la transformée de Fourier en dimension finie, l'impact de l'échantillonage sur le spectre du signal à traiter est étudié. Les bases et frames sur des espaces de fonctions sont ensuite introduites, en s'attardant sur le cas particulier des ondelettes. Le lien entre la transformée en ondelettes et les bancs de filtres est ensuite présenté. Dans ce cadre, la transformation polyphase et la factorisation lifting sont détaillées, en se préoccupant en particulier du cas uni ou bidimensionnel et des bancs de filtres à deux canaux.

Nous présentons également les résultats et outils de théorie de l'information utilisés pour la compression avec et sans perte. Nous commençons par la définition des diverses mesures d'information puis poursuivons en présentant la théorie débit-distorsion. Nous parlons enfin des techniques de codage arithmétique, de quantification scalaire et vectorielle, et d'allocation de débit par optimisation débit-distorsion utilisées dans la plupart des codeurs d'images actuels.

1.1 Théorie des lattices et échantillonnage

Afin de travailler aisément sur des signaux discrets de dimension finie d, nous nous plaçons dans le cadre de la théorie des lattices. Le terme lattice utilisé dans cette thèse désigne un réseau régulier de points. Nous nous contentons ici d'une description brève se limitant aux outils utiles à cette thèse, de plus amples détails pouvant être trouvés dans les ouvrages [1] et [2]. Les preuves des résultats énoncés se trouvent dans l'annexe 5.10.

1.1.1 Généralités sur les lattices

Soit $\mathsf{A} \in \mathcal{M}_d(\mathbb{R})$. Le produit d'un ensemble $\mathcal{B} \subset \mathbb{R}^d$ par la matrice A étant noté

$$\mathsf{A}\mathcal{B} = \{\mathsf{x} \in \mathbb{R}^d, \mathsf{x} = \mathsf{A}\mathsf{b}, \mathsf{b} \in \mathcal{B}\},$$

on définit la *lattice* issue de A par $\mathsf{A}\mathbb{Z}^d$.

Il s'agit d'une formalisation du domaine de définition du signal à traiter. Les lattices possèdent des caractéristiques particulières permettant de généraliser certains résultats obtenus en dimension 1. Cette section s'attache à présenter certaines de ces propriétés qui seront utiles par la suite.

Proposition 1. *Une lattice possède une structure de groupe additif commutatif.*

On appelle *sous-lattice* de $A\mathbb{Z}^d$ un sous-groupe de la lattice $A\mathbb{Z}^d$. La structure de groupe permet de généraliser les résultats à suivre au moyen de la propriété suivante :

Proposition 2. $\forall \mathsf{A} \in \boldsymbol{GL}_d(\mathbb{R}), \mathsf{A}\mathbb{Z}^d$ *est isomorphe à* $\mathbb{Z}^d$.

Ainsi, tout résultat valable sur la lattice $\mathbb{Z}^d$ se transpose sur une lattice quelconque $\mathsf{A}\mathbb{Z}^d$ dans la mesure où A est inversible. Notons qu'il peut exister plusieurs matrices génératrices de la même lattice.

Remarque 1. *La cellule élémentaire* $\mathsf{A}[0,1[^d$ *a un volume valant* $|\det(\mathsf{A})|$.

Exemple 1: Posons

$$\mathsf{Q} = \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}.$$

Cette matrice génère une lattice quinconce en dimension deux. Les vecteurs colonne de cette matrice forment une base de cette lattice. Le volume de la cellule élémentaire associée vaut 2 (Fig. 1.1). Cette même lattice est également issue de la matrice :

$$\mathsf{Q}' = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

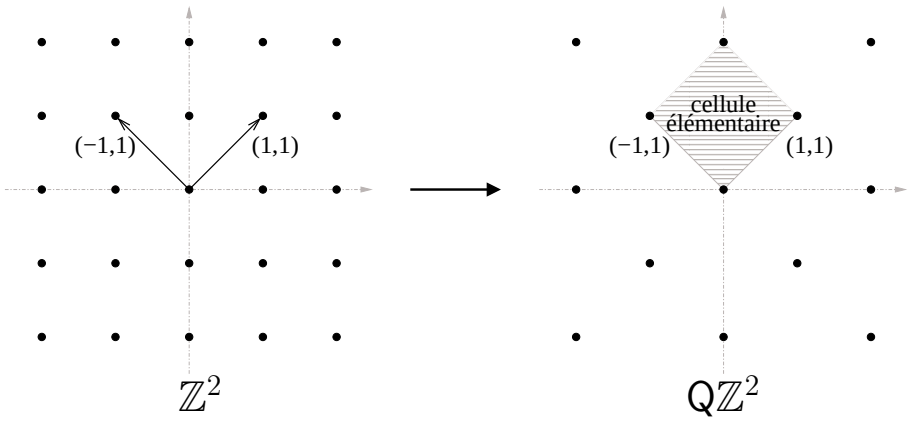

FIG. 1.1: Exemple de lattice quinconce et de sa cellule élémentaire associée.

1.1.2 Séparation en classes d'équivalence

Soit $\mathsf{M}\mathbb{Z}^d$ une sous-lattice de $\mathbb{Z}^d$ avec $\mathsf{M} \in \mathcal{M}_d(\mathbb{Z})$ matrice entière. On appelle translation de $\mathsf{M}\mathbb{Z}^d$ par le vecteur i l'ensemble $\mathsf{M}\mathbb{Z}^d + \mathsf{i} = \{\mathsf{i} + \mathsf{m}, \mathsf{m} \in \mathsf{M}\mathbb{Z}^d\}$. Cette translation définit une classe d'équivalence (ou *coset*) de la lattice $\mathsf{M}\mathbb{Z}^d$. Le vecteur $\mathsf{i} \in \mathbb{Z}^d$ est appelé *représentant* de cette classe d'équivalence.

On définit alors l'ensemble $\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d$, quotient de $\mathbb{Z}^d$, par $\mathsf{M}\mathbb{Z}^d$ par l'ensemble des classes d'équivalences de $\mathsf{M}\mathbb{Z}^d$ dans $\mathbb{Z}^d$. La notation $[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]$ désigne un ensemble de représentants des classes d'équivalences, comprenant exactement un représentant par classe. Cette notation n'est toutefois pas unique car n'importe quel élément d'une classe peut la représenter.

Proposition 3. ***(Division euclidienne)*** *Soit* $\mathsf{M} \in \mathcal{M}_d(\mathbb{Z}), \det(\mathsf{M}) \neq 0$ *une matrice carrée entière de déterminant non nul. Soit l'ensemble* $\mathcal{R}^d_{\mathsf{M}} = \mathbb{Z}^d \cap \mathsf{M}[0,1[^d$. *Alors*

$$\forall \mathsf{z} \in \mathbb{Z}^d, \exists ! \mathsf{q} \in \mathbb{Z}^d, \mathsf{r} \in R^d_{\mathsf{M}}, \mathsf{z} = \mathsf{M}\mathsf{q} + \mathsf{r}.$$

Corrolaire 1. $\mathcal{R}^d_{\mathsf{M}}$ *est un ensemble de représentants de* $\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d$. *On notera par la suite de manière unique* $[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d] = \mathcal{R}^d_{\mathsf{M}}$.

La séparation d'une lattice $\mathbb{Z}^d$ en cosets consiste alors à partitionner $\mathbb{Z}^d$ en classes d'équivalence d'une lattice $\mathsf{M}\mathbb{Z}^d$:

$$\mathbb{Z}^d = \mathsf{M}\mathbb{Z}^d + [\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d].$$

Exemple 2: Reprenons la matrice Q *définie dans l'exemple 1. Le plan discret* $\mathbb{Z}^2$ *est partitionné en deux classes d'équivalence (Fig. 1.2) :*

$$\mathbb{Z}^2 = \mathsf{Q}\mathbb{Z}^2 + [\mathbb{Z}^2/\mathsf{Q}\mathbb{Z}^2],$$

où

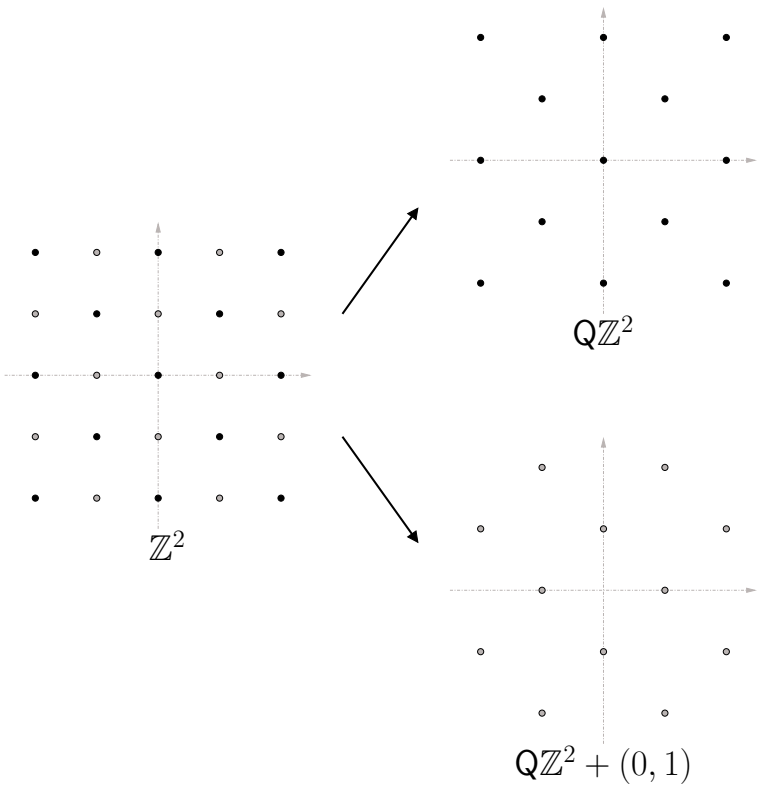

FIG. 1.2: Séparation de $\mathbb{Z}^2$ en deux lattices quinconces complémentaires.

$$[\mathbb{Z}^2/\mathsf{Q}\mathbb{Z}^2] = \{(0,0)^\top; (0,1)^\top\}.$$

1.1.3 Échantillonnage

On considère dans cette section un signal x discret de dimension finie d défini sur $\mathbb{Z}^d$ et à valeurs réelles :

$$\begin{aligned} x : \mathbb{Z}^d &\to \mathbb{R} \\ \mathsf{n} &\mapsto x[\mathsf{n}]. \end{aligned}$$

Le signal $x_{\downarrow\mathsf{M}}[n]$ sous-échantillonné de x par M est obtenu à partir du signal x en ne conservant que les échantillons situés sur la lattice issue de M :

$$x_{\downarrow\mathsf{M}}[\mathsf{n}] = x[\mathsf{M}\mathsf{n}].$$

Le signal $x_{\uparrow\mathsf{M}}[n]$ sur-échantillonné de x par M inversible est défini sur la lattice issue de M à partir des échantillons de x. En dehors de cette lattice un remplissage par des

zéros est effectué :

$$x_{\uparrow \mathsf{M}}[\mathsf{n}] = \begin{cases} x[\mathsf{M}^{-1}\mathsf{n}] & si\ \mathsf{n} \in \mathsf{M}\mathbb{Z}^d \\ 0 & sinon. \end{cases}$$

On a donc les propriétés suivantes :

$$x_{\uparrow \mathsf{M}\downarrow \mathsf{M}}[\mathsf{n}] = x[\mathsf{n}]$$

$$x_{\downarrow \mathsf{M}\uparrow \mathsf{M}}[\mathsf{n}] = \begin{cases} x[\mathsf{n}] & si\ \mathsf{n} \in \mathsf{M}\mathbb{Z}^d \\ 0 & sinon. \end{cases}$$

1.1.4 Transformée de Fourier d'un signal discret multidimensionnel

1.1.4.1 Définition

La transformée de Fourier d'un signal discret en dimension d est définie par :

$$\forall \mathsf{f} \in \mathbb{R}^d, X(\mathsf{f}) = \sum_{\mathsf{n} \in \mathbb{Z}^d} x[\mathsf{n}] e^{-2i\pi \mathsf{f}^\top \mathsf{n}},$$

où $\mathsf{f}^\top \mathsf{n}$ est le produit scalaire entre f et n. Le module de cette transformée donne la *réponse fréquentielle* ou le *spectre* du signal multidimensionnel.

Proposition 4. ***(Somme des racines de l'unité)*** *Soit* $\mathsf{M} \in \mathcal{M}_d(\mathbb{Z}), \det(\mathsf{M}) \neq 0$, *alors*

$$\sum_{\mathsf{x} \in \mathcal{R}_{\mathsf{M}}^d} e^{2i\pi (\mathsf{M}^{-1}\mathsf{t})^\top \mathsf{x}} = \begin{cases} |\det(\mathsf{M})| & si\ \mathsf{t} \in \mathsf{M}\mathbb{Z}^d \\ 0 & sinon. \end{cases}$$

Ce résultat généralise la propriété de la somme des racines de l'unité aux dimensions finies. Elle permet d'étudier l'impact du sous-échantillonage sur le spectre d'un signal multidimensionnel.

Corrolaire 2. *Le cardinal de* $[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]$ *vaut* $|\det(\mathsf{M})|$.

1.1.4.2 Impact de l'échantillonnage dans le domaine fréquentiel

Proposition 5. ***(sur-échantillonnage)*** *La transformée de Fourier d'un signal sur-échantillonné par* M *s'exprime en fonction de la transformée de Fourier du signal original par :*

$$X_{\uparrow \mathsf{M}}(\mathsf{f}) = X(\mathsf{M}^\top \mathsf{f}). \tag{1.1}$$

Proposition 6. ***(sous-échantillonnage)*** *La transformée de Fourier d'un signal sous-échantillonné par* M *s'exprime en fonction de la transformée de Fourier du signal original par :*

$$X_{\downarrow \mathsf{M}}(\mathsf{f}) = \frac{1}{|\det(\mathsf{M})|} \sum_{\mathsf{k} \in [\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]} X(\mathsf{M}^{-\top}(\mathsf{f} - \mathsf{k})). \tag{1.2}$$

Corrolaire 3. ***(repliement de spectre)*** *Le sous-échantillonnage suivi du sur-échantillonnage d'un signal x entraîne une réplication de son spectre fréquentiel selon la lattice issue de $\mathsf{M}^{-\top}$, appelée lattice réciproque de $\mathsf{M}\mathbb{Z}^d$:*

$$X_{\downarrow \mathsf{M} \uparrow \mathsf{M}}(\mathsf{f}) = \frac{1}{|\det(\mathsf{M})|} \sum_{\mathsf{k} \in [\mathbb{Z}^d / \mathsf{M}\mathbb{Z}^d]} X(\mathsf{f} - \mathsf{M}^{-\top}\mathsf{k})$$

Ce résultat généralise la formule du repliement de spectre en dimension 1. On rappelle que la transformée de Fourier d'un peigne de Dirac de pas Δ est également un peigne de Dirac, dont le pas est $1/\Delta$. Ainsi, un sous-échantillonnage suivi d'un sur-échantillonnage d'un facteur Δ entraîne une réplication du spectre du signal tous les $1/\Delta$. Dans le cas général, on remarque qu'un sous-échantillonnage par une lattice de cellule élémentaire de volume $|\det(M)|$ suivi du sur-échantillonnage correspondant se traduit par une réplication du spectre sur une lattice dont le volume de la cellule élémentaire est inversement proportionnel.

Exemple 3: Intéressons-nous à nouveau au cas particulier des lattices quinconces en dimension 2 en poursuivant l'exemple 2 sur la séparation de $\mathbb{Z}^2$ en grilles quinconces.

La lattice réciproque de $Q\mathbb{Z}^2$ est définie par la matrice :

$$\mathsf{Q}^{-\top} = -\frac{1}{2}\mathsf{Q} = \begin{pmatrix} -\frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{pmatrix}.$$

Après sous-échantillonnage et sur-échantillonnage par Q on obtient donc une réplication du spectre selon cette lattice, comme

$$Q^{-\top}[\mathbb{Z}^2/\mathsf{Q}\mathbb{Z}^2] = \{0, \left(-\frac{1}{2}, \frac{1}{2}\right)^\top\}$$

Le spectre du signal ré-échantillonné s'exprime donc de la manière suivante :

$$X_{\downarrow \mathsf{M} \uparrow \mathsf{M}}(\mathsf{f}) = \frac{1}{2}\left(X(\mathsf{f}) + X(\mathsf{f} - \left(-\frac{1}{2}, \frac{1}{2}\right)^\top)\right).$$

On constate donc une réplication du spectre du signal selon une configuration quinconce également (Fig. 1.3).

1.2 Bases et frames

1.2.1 Représentation vectorielle des signaux

Par la suite, nous représentons les signaux traités par des fonctions discrètes d'énergie finie définies sur un domaine $\mathbb{I} \subseteq \mathbb{Z}^d$. En particulier, nous nous intéressons aux images et à leur contours, qui correspondent à des cas particuliers de signaux bi- et uni-dimensionnels. L'ensemble de ces fonctions, noté $\mathcal{L}_2(\mathbb{I})$ forme un espace vectoriel de dimension $0 \leq L \leq \infty$, où le produit scalaire est défini par :

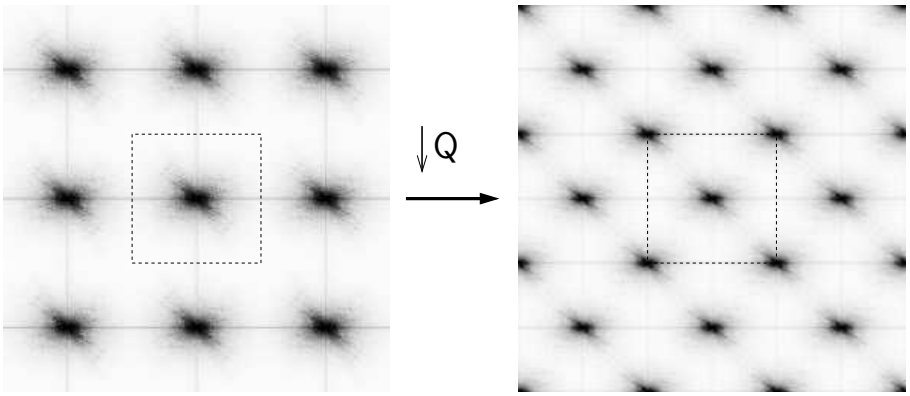

FIG. 1.3: Repliement du spectre de l'image *lena* [3] dû au ré-échantillonnage sur grille quinconce. La zone en pointillé représente la zone du spectre considérée par la transformée de Fourier discrète.

$$\langle f, g \rangle = \sum_{x \in \mathbb{I}} f[x] g[x].$$

De plus on a pour toute image :

$$\|f\|^2 = \langle f, f \rangle = \sum_{x \in \mathbb{I}} f[x]^2 < \infty.$$

Une frame $\mathcal{F} = (\psi_k)_{k \in [\![0, K-1]\!]}$, avec $L \leq K \leq \infty$ et $\forall k \in [\![0, K-1]\!], \psi_k \in \mathcal{L}_2(\mathbb{I})$, est une famille génératrice de l'espace $\mathcal{L}_2(\mathbb{I})$ telle que

$$\forall f \in \mathcal{L}_2(\mathbb{I}), A\|f\|^2 \leq \sum_{k=0}^{K-1} |\langle f, \psi_k \rangle|^2 \leq B\|f\|^2,$$

avec $0 < A \leq B < \infty$. La frame est dite *étroite* lorsque $A = B$. Si de plus $A = B = 1$, on retrouve la propriété de conservation de l'énergie exprimée par la relation de Parseval :

$$\forall f \in \mathcal{L}_2(\mathbb{I}), \sum_{k=0}^{K-1} |\langle f, \psi_k \rangle|^2 = \|f\|^2.$$

Une base est une frame qui est en outre une famille libre. Une base étroite est orthogonale. Elle est orthonormée si $A = B = 1$.

Si $\mathbb{I}$ est de dimension finie et si $\mathcal{F}$ est une base alors $K = L$. En d'autres termes, dans le cas d'une image restreinte à un domaine borné, le nombre de vecteurs de base nécessaires et suffisants pour représenter l'image est égal au nombre de pixels de l'image. On parle alors de représentation *à échantillonnage critique*, par opposition à une représentation *redondante* dans le cas où $\mathcal{F}$ est une famille liée.

On appelle frame *duale* de $\mathcal{F}$ la frame $\mathcal{F}^\star = (\psi_k^\star)_{k \in [\![0, K-1]\!]}$ vérifiant :

$$\forall f \in \mathcal{L}_2(\mathbb{I}), f = \sum_{k=0}^{K-1} \langle f, \psi_k \rangle \, \psi_k^\star = \sum_{k=0}^{K-1} \langle f, \psi_k^\star \rangle \, \psi_k.$$

1.2.2 Opérateurs d'analyse et de synthèse

Soit A l'opérateur linéaire *d'analyse* défini de $\mathcal{L}_2(\mathbb{I})$ dans $\mathbb{R}^K$ par

$$\forall f \in \mathcal{L}_2(\mathbb{I}), \mathsf{A}f = \sum_{k=0}^{K-1} \langle \psi_k, f \rangle \, e_k,$$

où $(e_k)_{k \in [\![0,K-1]\!]}$ est une base orthonormée de $\mathbb{R}^K$.

Il existe un opérateur *pseudo-inverse* de A appelé opérateur de *synthèse* et noté $\mathsf{A}^\perp$. Cet opérateur linéaire est défini de $\mathbb{R}^K$ dans $\mathcal{L}_2(\mathbb{I})$ par

$$\forall x \in \mathbb{R}^K, \mathsf{A}^\perp x = \sum_{k=0}^{K-1} \langle x, e_k \rangle \, \psi_k^\star.$$

On obtient de manière immédiate que $\mathsf{A}^\perp \mathsf{A} f = f$. Par contre $\mathsf{A}\mathsf{A}^\perp f = f$ si et seulement si $\mathcal{F}$ et $\mathcal{F}^\star$ sont des bases. Par construction de la base duale on a $\forall i, j \in [\![0, K-1]\!], i \neq j, \left\langle \psi_i, \psi_j^\star \right\rangle = 0$. On dit alors que $\mathcal{F}$ et $\mathcal{F}^\star$ forment un système de bases biorthogonales. Si $\mathcal{F}$ est orthogonale alors $\mathcal{F} = \mathcal{F}^\star$.

Dans le cas où les deux espaces $\mathbb{R}^K$ et $\mathcal{L}_2(\mathbb{I})$ sont de dimension finie, l'opérateur d'analyse est associable à une matrice $\mathsf{A} \in \mathcal{M}_{L,K}(\mathbb{R})$. On obtient la matrice associée à l'opérateur inverse en calculant la matrice $\mathsf{A}^\perp \in \mathcal{M}_{K,L}(\mathbb{R})$ pseudo-inverse de A définie comme :

$$\mathsf{A}^\perp = (\mathsf{A}^\top \mathsf{A})^{-1} \mathsf{A}^\top.$$

Si de plus $\mathcal{F}$ et $\mathcal{F}^\star$ sont des bases, on a

$$\mathsf{A}^\perp = \mathsf{A}^{-1}.$$

Enfin si $\mathcal{F}$ est orthonormée,

$$\mathsf{A}^\perp = \mathsf{A}^\top.$$

L'opérateur de synthèse a la propriété de minimiser l'énergie du bruit introduit sur le signal reconstruit lorsque qu'un bruit indépendant est ajouté au signal transformé [4].

Exemple 4: Considérons la base formée des vecteurs $\mathsf{e}_0 = (0,1)^\top$ *et* $\mathsf{e}_1 = \left(\frac{\sqrt{3}}{2}, -\frac{1}{2}\right)^\top$. *L'opérateur d'analyse associé à cette base s'écrit :*

$$\mathsf{A} = \begin{pmatrix} 0 & 1 \\ \frac{\sqrt{3}}{2} & -\frac{1}{2} \end{pmatrix}.$$

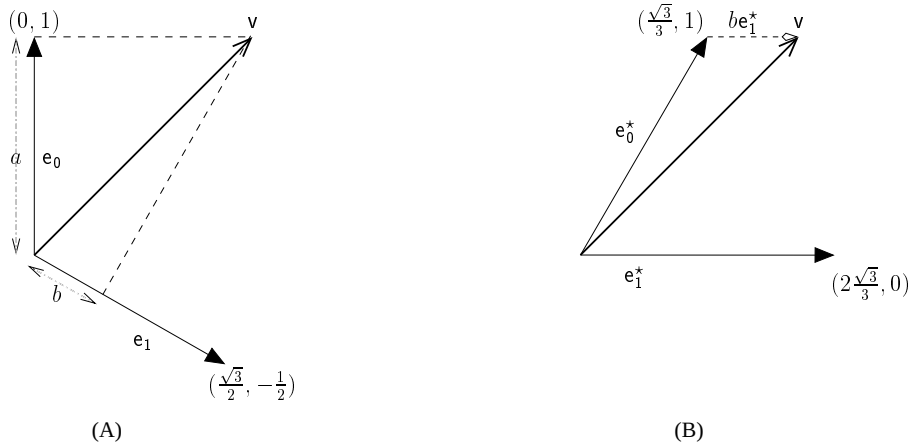

FIG. 1.4: Exemple de bases biorthogonales. (A) Base de $\mathbb{R}^2$. (B) Base duale.

Comme $(\mathsf{e}_0, \mathsf{e}_1)$ *est une base finie, l'opérateur de synthèse se déduit de la matrice inverse de* A *:*

$$\mathsf{A}^{\perp} = \mathsf{A}^{-1} = \begin{pmatrix} \frac{\sqrt{3}}{3} & 2\frac{\sqrt{3}}{3} \\ 1 & 0 \end{pmatrix}.$$

La base duale de $(\mathsf{e}_0, \mathsf{e}_1)$ *s'obtient alors comme les vecteurs colonne de* $\mathsf{A}^{\perp}$*. Ainsi* $\mathsf{e}_0{}^{\star} = \left(\frac{\sqrt{3}}{3}, 1\right)^{\top}$ *et* $\mathsf{e}_1{}^{\star} = \left(2\frac{\sqrt{3}}{3}, 0\right)^{\top}$*. On constate qu'on a bien* $\langle \mathsf{e}_0, \mathsf{e}_1{}^{\star} \rangle = 0$ *et* $\langle \mathsf{e}_1, \mathsf{e}_0{}^{\star} \rangle = 0$*. Les bases* $(\mathsf{e}_0, \mathsf{e}_1)$ *et* $(\mathsf{e}_0{}^{\star}, \mathsf{e}_1{}^{\star})$ *sont donc bien biorthogonales (Fig. 1.4).*

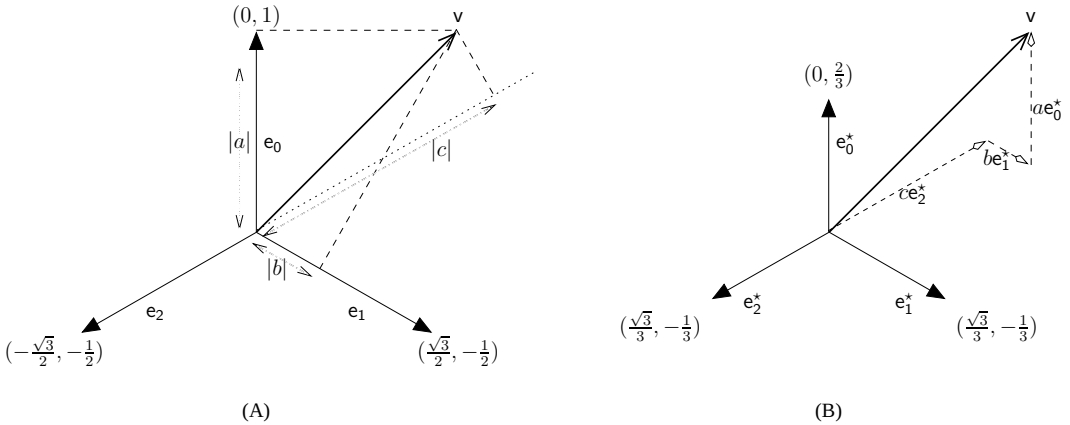

FIG. 1.5: Exemple de frames duales.

Exemple 5: Ajoutons le vecteur $\mathsf{e}_2 = \left(-\frac{\sqrt{3}}{2}, -\frac{1}{2}\right)^{\top}$ *à la base précédente pour obtenir la*

frame $(\mathsf{e}_0, \mathsf{e}_1, \mathsf{e}_2)$ *(Fig. 1.5 A). L'opérateur d'analyse associé à cette frame s'écrit alors :*

$$\mathsf{A} = \begin{pmatrix} 0 & 1 \\ \frac{\sqrt{3}}{2} & -\frac{1}{2} \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} \end{pmatrix}.$$

On obtient cette fois ci l'opérateur de synthèse en calculant la matrice pseudo inverse de A *:*

$$\mathsf{A}^{\perp} = (\mathsf{A}^{\top}\mathsf{A})^{-1}\mathsf{A}^{\top} = \begin{pmatrix} 0 & \frac{\sqrt{3}}{3} & -\frac{\sqrt{3}}{3} \\ \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} \end{pmatrix} = \frac{2}{3}\mathsf{A}^{\top}.$$

La frame duale de $(\mathsf{e}_0, \mathsf{e}_1, \mathsf{e}_2)$ *s'obtient de même en lisant les vecteurs colonne de* $\mathsf{A}^{\perp}$*. Elle reconstruit bien tout vecteur* v *par addition dans cette frame duale des contributions de chaque projection (Fig.1.5 B). Par contre, contrairement au cas des bases,* $\langle \mathsf{e}_0, {\mathsf{e}_1}^{\star} \rangle \neq 0$ *et* $\langle \mathsf{e}_0, {\mathsf{e}_2}^{\star} \rangle \neq 0$*.*

1.2.3 Espace noyau et image d'une frame

Dans le cas où une frame est redondante, l'espace transformé peut se séparer en deux espaces complémentaires *image* et *noyau* de l'opérateur linéaire d'analyse. Il existe en effet, du fait de la redondance, plusieurs représentations possible d'un même signal dans le domaine transformé. On peut ainsi ajouter n'importe quel vecteur de l'espace noyau à la représentation transformée du signal sans modifier le signal reconstruit après transformée inverse. Il est donc intéressant de choisir une représentation particulière dans l'espace transformé suivant le critère que l'on cherche à optimiser (par exemple un coût de codage minimal). Nous noterons par la suite ces espaces image et noyau $Im(\mathsf{A})$ et $Ker(\mathsf{A})$ respectivement, avec $Ker(\mathsf{A}) = \varnothing$ lorsque la frame est une base.

1.2.4 Ondelettes

Les ondelettes sont un cas particulier de familles de fonctions de $\mathcal{L}_2(\mathbb{Z}^d)$. Nous nous intéressons plus particulièrement aux bases d'ondelettes en dimension finie, et principalement aux cas uni- et bi-dimensionnel.

Soit un signal discret d'énergie finie $x \in \mathcal{L}_2(\mathbb{Z}^d)$ à valeurs réelles. Soit une matrice entière M telle que $|\det \mathsf{M}| > 1$. La transformée en ondelettes consiste à représenter x par sa décomposition sur une famille de fonctions d'ondelettes :

$$\mathcal{W} = \left(\Psi^m_{l,\mathsf{n}}\right)_{l\in\mathbb{N},\mathsf{n}\in\mathbb{Z}^d,m\in[\![1,|\det \mathsf{M}|-1]\!]},$$

$$\forall \mathsf{t} \in \mathbb{Z}^d, \Psi^m_{l,\mathsf{n}}[\mathsf{t}] = |\det \mathsf{M}|^{-\frac{l}{2}} \Psi^m\left(\mathsf{M}^{-l}\mathsf{t} - \mathsf{n}\right).$$

Les fonctions Ψ^m sont appelées *fonctions d'ondelette* ou *ondelettes mères*, tandis que les fonctions $\Psi^m_{l,\mathsf{n}}$ représentent des versions dilatées et translatées de ces fonctions. La variable l indique le *niveau de décomposition* (ou l'*échelle*) tandis que n indique la

position de $\Psi^m_{l,\mathsf{n}}$. L'exposant m indique de quelle fonction d'ondelette $\Psi^m_{l,\mathsf{n}}$ est issue ; il sera omis lorsqu'il n'y a pas d'ambiguïté. Par la suite, nous utiliserons la notation $y^m_{l,\mathsf{n}}$ pour désigner le coefficient d'ondelette m correspondant à l'échelle l et la position n :

$$y^m_{l,\mathsf{n}} = \left\langle \Psi^m_{l,\mathsf{n}}, x \right\rangle = \sum_{\mathsf{t}\in\mathbb{Z}^d} \Psi^m_{l,\mathsf{n}}[\mathsf{t}]x[\mathsf{t}].$$

Pour des raisons pratiques on préfère en général avoir une décomposition sur un nombre fini L de niveaux. On définit alors les fonctions suivantes lorsqu'elles existent :

$$\Phi_{L,\mathsf{n}}(\mathsf{t}) = \sum_{l=L+1}^{\infty} \sum_{m=1}^{|\det \mathsf{M}|-1} \Psi^m_{l,\mathsf{n}}[\mathsf{t}],$$

Ces fonctions fournissent alors une décomposition sur la famille :

$$\overline{\mathcal{W}_L} = \left(\Psi^m_{l,\mathsf{n}}, \Phi_{L,\mathsf{n}}\right)_{l\in[\![1,L]\!],\mathsf{n}\in\mathbb{Z}^d,m\in[\![1,|\det \mathsf{M}|-1]\!]}.$$

La fonction $\Phi_{0,\mathsf{0}}$ est appelée *fonction d'échelle*. Toute autre fonction $\Phi_{L,\mathsf{n}}$ peut s'en déduire par translation ou dilatation. Notons que cette fonction d'échelle étant définie comme une série infinie, elle n'existe pas forcément pour toute fonction d'ondelettes.

Une fonction d'ondelette doit être à support borné et de somme nulle :

$$\exists M \in \mathbb{R}, \forall \mathsf{t} \in \mathbb{R}^d, \|\mathsf{t}\| > M, \Psi(\mathsf{t}) = 0$$

$$\int_{\mathbb{R}^d} \Psi(\mathsf{t}) d\mathsf{t} = 0.$$

Elle peut être choisie de diverses manières en fonction des propriétés désirées sur la base de décomposition. Parmi les principales propriétés on retiendra :

1. **la reconstruction parfaite** : la projection de x sur la famille $\mathcal{W}$ doit être inversible. Cette condition impose que $\mathcal{W}$ forme une frame de $\mathcal{L}_2(\mathbb{Z}^d)$. Dans ce cas il existe une frame d'ondelette duale $\mathcal{W}^\star = \left(\Psi^{m\star}_{l,\mathsf{n}}\right)_{l\in\mathbb{N},\mathsf{n}\in\mathbb{Z}^d,m\in[\![1,|\det \mathsf{M}|-1]\!]}$ telle que :

$$\forall \mathsf{t} \in \mathbb{Z}^d, x[\mathsf{t}] = \sum_{m=1}^{|\det M|-1} \sum_{l=0}^{\infty} \sum_{\mathsf{n}\in\mathbb{Z}^d} y^m_{l,\mathsf{n}} \Psi^{m\star}_{l,\mathsf{n}}[\mathsf{t}]$$

2. **l'orthogonalité** : les fonctions de base $\mathcal{W}$ sont orthogonales :

$$\forall (l,\mathsf{n},m) \neq (l',\mathsf{n}',m'), \sum_{\mathsf{t}\in\mathbb{Z}^d} \Psi^m_{m,\mathsf{n}}[\mathsf{t}]\Psi^{m'}_{l',\mathsf{n}'}[\mathsf{t}] = 0.$$

La base duale est alors identique à la base de décomposition, i.e. $\mathcal{W} = \mathcal{W}^*$. Si la base est de plus orthonormée, l'énergie de x est conservée :

$$\sum_{m=1}^{|\det M|-1} \sum_{l=0}^{\infty} \sum_{\mathsf{n}\in\mathbb{Z}^d} \|y^m_{l,\mathsf{n}}\|^2 = \sum_{\mathsf{t}\in\mathbb{Z}^d} x[\mathsf{t}]^2.$$

3. **la phase linéaire** : les fonctions d'ondelette mère sont symétriques ou anti-symétriques :

$$\forall m \in [\![1, |\det M| - 1]\!], \exists \mathsf{T} \in \mathbb{Z}^d, \forall \mathsf{t} \in \mathbb{Z}^d, \Psi^m[t] = \pm \Psi^m[\mathsf{T} - \mathsf{t}]$$

4. **la régularité (d'ordre P, selon h)** : la projection sur une fonction d'ondelette mère de tout polynôme d'ordre inférieur ou égal à P orienté selon $\mathsf{h} \neq 0$ est nulle :

$$\forall p \in [\![1, P]\!], \int_{\mathbb{R}} t^p \mathsf{h} \Psi(\mathsf{t}) dt = 0.$$

En traitement d'image, la propriété de reconstruction parfaite est essentielle pour ne pas dégrader le signal lors de son traitement et former une bijection entre le domaine transformé et le domaine spatial. La propriété de phase linéaire évite les distorsions de phase très désagréables visuellement (délocalisation des contours). La symétrie des filtres peut par ailleurs être exploitée afin de réduire la complexité des traitements. Enfin, cette propriété permet une réalisation aisée de bases d'ondelettes définies sur un domaine limité par périodisation symétrique du signal en dehors du domaine. On cherche à approcher le plus possible de la propriété d'orthogonalité afin que la distorsion introduite par le traitement dans le domaine transformé se retrouve dans le domaine spatial après recomposition. Une régularité forte de l'ondelette est enfin essentielle pour obtenir de bonnes propriétés de décorrélation du signal.

Cependant il est impossible de satisfaire toutes ces conditions. L'ondelette de Haar définie par :

$$\begin{cases} \Psi(t) = & \frac{1}{\sqrt{2}} & \text{si } t \in [-1, 0[\\ \Psi(t) = & -\frac{1}{\sqrt{2}} & \text{si } t \in [0, 1[\\ \Psi(t) = & 0 & \textit{sinon.} \end{cases}$$

est la seule à pouvoir satisfaire les 3 premières conditions à la fois en dimension 1 mais possède une régularité nulle.

Les ondelettes unidimensionnelles servent souvent de base à l'élaboration d'ondelettes en dimension plus élevée. Dans cette thèse, nous nous intéressons principalement aux deux cas particuliers que sont l'ondelette séparable dyadique et l'ondelette quinconce.

1.2.4.1 Ondelette séparable dyadique

Étant donné une fonction d'ondelette unidimensionnelle Ψ et sa fonction d'échelle associée Φ, les fonctions de base suivantes sont définies de manière séparable :

$$\begin{cases} \Phi^{LL}(u, v) = \Phi(u)\Phi(v) \\ \Psi^{HL}(u, v) = \Psi(u)\Phi(v) \\ \Psi^{LH}(u, v) = \Phi(u)\Psi(v) \\ \Psi^{HH}(u, v) = \Psi(u)\Psi(v) \end{cases}$$

Le signal bidimensionnel x est alors décomposé sur la frame :

$$\left(\Psi^{HL}_{l,\mathsf{n}}, \Psi^{LH}_{l,\mathsf{n}}, \Psi^{HH}_{l,\mathsf{n}}, \Phi^{LL}_{L,\mathsf{n}}\right)_{l\in[\![0,L]\!],\mathsf{n}\in\mathbb{Z}^2}.$$

On obtient ainsi une décomposition sur une frame formée de trois ondelettes différentes et d'une fonction d'échelle. Le facteur de dilatation entre chaque échelle vaut $|\det 2I|^{\frac{1}{2}} = 2$ sur chaque axe. L'intérêt de cette décomposition est d'être réalisable en effectuant tout d'abord une décomposition des lignes de x selon la frame d'ondelette unidimensionnelle, puis en appliquant au résultat une décomposition des colonnes (ou inversement). On se ramène donc aisément au cas unidimensionnel en remarquant que la transformée complète peut s'écrire

$$(\mathsf{A}(\mathsf{A}x)^\top)^\top = \mathsf{A}x\mathsf{A}^\top,$$

où A est l'opérateur d'analyse associé à la frame d'ondelette unidimensionnelle.

Les propriétés de reconstruction parfaite, d'orthogonalité et de phase linéaire sont conservées. Pour une ondelette unidimensionnelle régulière d'ordre P, les ondelettes Ψ^{LH} et Ψ^{HL} sont également régulières d'ordre P *selon une orientation verticale ou horizontale uniquement*. De même, Ψ^{HH} est régulière d'ordre P selon des directions verticale *et* horizontale. Leur régularité est en revanche nulle dans toute autre direction. Ce dernier point motive la recherche d'ondelettes ayant une forte régularité dans un plus grand nombre d'orientations, afin de représenter efficacement les contours des images.

1.2.4.2 Ondelette quinconce

L'analyse en ondelettes quinconces [5] ne nécessite qu'une seule fonction d'ondelette pour représenter un signal bidimensionnel x. La relation liant les différentes échelles de l'ondelette est définie par la matrice quinconce Q (Ex. 2). Le facteur de dilatation entre chaque échelle est alors donné par $|\det \mathsf{Q}|^{\frac{1}{2}} = \sqrt{2}$.

Il est possible d'obtenir de telles ondelettes à partir d'ondelettes unidimensionnelles en effectuant une transformation de McClellan [6]. Il existe également des procédés similaires effectuant des transformations directement sur les composantes polyphases [7] ou les pas lifting [8] de l'ondelette unidimensionnelle. Ces transformations conservent également les propriétés de phase linéaire et de reconstruction parfaite de l'ondelette unidimensionnelle. Notons que les propriétés de régularité et d'orthogonalité ne sont pas nécessairement conservées par ce type de transformations.

1.3 Analyse en sous-bandes

1.3.1 Bancs de filtres

1.3.1.1 Décomposition en sous-bandes

Une transformée en ondelettes discrète est réalisable au moyen d'un banc de filtres à réponse impulsionnelle finie itéré sur la bande basse (Fig. 1.7). Le signal reconstruit est obtenu en sortie du banc de filtres dual par sommation des sorties de chaque filtre (Fig.

1.6). Dans le cas général, le nombre de canaux de ce banc de filtres est égal à $|\det(\mathsf{M})|$, où M est la matrice de sous-échantillonnage. Ainsi le nombre d'échantillons filtrés en sortie du banc de filtres est identique au nombre d'échantillons en entrée. Cependant, nous nous intéressons ici uniquement au cas des bancs de filtres à deux canaux. Ce cadre n'est pas trop limitant car les bancs de filtres à canaux multiples étudiés par la suite sont tous décomposables en plusieurs bancs de filtres à deux canaux appliqués successivement. Nous évoquons ici uniquement les résultats principaux de la théorie des bancs de filtres, une présentation plus détaillée étant disponible dans l'ouvrage [2].

Un banc de filtres d'analyse en ondelette est constitué d'une paire de filtres, l'un passe bas (H0), l'autre passe haut (H1). La réponse impulsionnelle du filtre passe haut correspond à la fonction d'ondelette mère tandis que celle du filtre passe bas correspond à la fonction d'échelle associée. Après L itérations du banc de filtres réalisant l'analyse en ondelettes, on obtient en sortie de chaque canal la projection de x sur la base $(\Psi_{l,\mathsf{n},m}, \Phi_{L,\mathsf{n},m})_{l\in[\![1,L]\!],\mathsf{n}\in\mathbb{Z}^d}$. La projection de x sur $(\Phi_{L,\mathsf{n}})_{\mathsf{n}\in\mathbb{Z}^d}$ correspond à la bande basse fréquence du signal pour cette décomposition. Pour un signal continu par morceaux, la majeure partie de l'énergie se retrouve dans cette bande après transformation. Les discontinuités entraînent des valeurs importantes de coefficients dans les autres bandes. Plus la régularité de l'ondelette est forte, plus l'énergie d'un signal corrélé aura tendance à se concentrer dans la sous-bande de basse fréquence, ce qui est l'objectif de la transformation dans les schémas de compression par transformée.

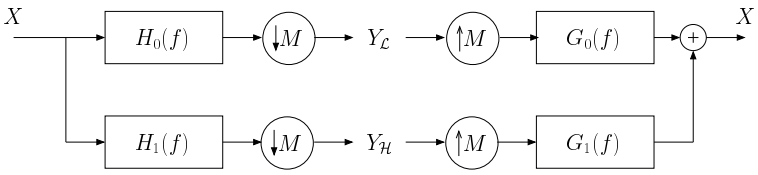

FIG. 1.6: Banc de filtres d'ondelettes à deux canaux.

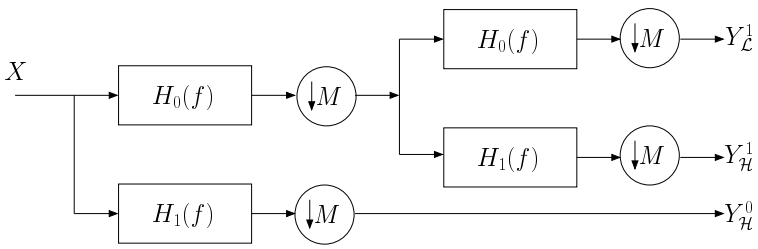

FIG. 1.7: Itération du banc de filtres sur la bande basse.

1.3.1.2 Propriétés du banc de filtres à deux canaux

La projection de x sur $\Psi_{l,n}$ s'obtient en sortie du filtre passe-haut H_1 au niveau l. La sortie du filtre passe-bas H_0 permet de réitérer l'analyse ou d'obtenir la projection

de x sur $\Phi_{L,n}$ au niveau final L. De même, les filtres passe-bas G_0 et passe-haut G_1 resynthétisent x à partir de ses projections.

La condition de reconstruction parfaite peut s'exprimer directement dans le domaine de Fourier sur les réponses fréquentielles des filtres d'analyse $H_0(f), H_1(f)$ et de synthèse $G_0(f), G_1(f)$.

Théorème 1. *Pour obtenir la reconstruction parfaite $\hat{X} = X$ avec un banc de filtres à deux canaux, il faut et il suffit que*

$$\begin{cases} H_0(\mathsf{f})G_0(\mathsf{f}) + H_1(\mathsf{f})G_1(\mathsf{f}) = 2 \\ H_0(\mathsf{f} - \mathsf{M}^{-\top}\mathsf{j})G_0(\mathsf{f}) + H_1(\mathsf{f} - \mathsf{M}^{-\top}\mathsf{j})G_1(\mathsf{f}) = 0 \end{cases} . \tag{1.3}$$

en notant j *l'unique élément non nul de* $[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]$.

Pour un banc de filtres à réponse impulsionnelle finie ce théorème mène à la condition suivante :

$$\begin{cases} G_0(\mathsf{f}) & = H_1(\mathsf{f} - \mathsf{M}^{-\top}\mathsf{j}) \\ G_1(\mathsf{f}) & = -H_0(\mathsf{f} - \mathsf{M}^{-\top}\mathsf{j}) \end{cases} . \tag{1.4}$$

Dans le cas d'un banc de filtres orthogonaux, on a de plus [9] :

$$\begin{cases} G_0(\mathsf{f}) = H_0(\mathsf{f}) \\ G_1(\mathsf{f}) = -H_1(\mathsf{f}) \end{cases} . \tag{1.5}$$

1.3.1.3 Matrice de modulation

On définit la *matrice de modulation* d'un banc de filtres H par :

$$\mathsf{H}(f) = \begin{pmatrix} H_0(\mathsf{f}) & H_0(\mathsf{f} - \mathsf{M}^{-\top}\mathsf{j}) \\ H_1(\mathsf{f}) & H_1(\mathsf{f} - \mathsf{M}^{-\top}\mathsf{j}) \end{pmatrix}.$$

La condition de reconstruction parfaite décrite par l'équation 1.3 s'obtient alors sous forme matricielle :

$$\mathsf{G}^{\top}\mathsf{H} = 2\mathsf{I}.$$

1.3.2 Lifting

1.3.2.1 Identités remarquables

On observe qu'il est possible d'intervertir le filtrage et le ré-échantillonnage au moyen des identités remarquables suivantes [2] :

En effet, en utilisant l'expression 1.1 on a pour le sur-échantillonnage (Fig. 1.8) :

$$Y(\mathsf{f}) = X_{\uparrow\mathsf{M}}(\mathsf{f})H(\mathsf{M}^{\top}\mathsf{f}) = X(\mathsf{M}^{\top}\mathsf{f})H(\mathsf{M}^{\top}\mathsf{f}) = [XH]_{\uparrow\mathsf{M}}(\mathsf{f}).$$

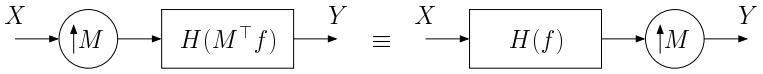

FIG. 1.8: Identités remarquables sur les bancs de filtres : sur-échantillonnage.

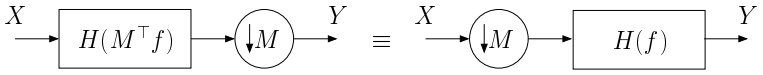

FIG. 1.9: Identités remarquables sur les bancs de filtres : sous-échantillonnage.

De même, en utilisant 1.2 pour le sous-échantillonnage (Fig. 1.9) on a :

$$\begin{aligned} Y(\mathsf{f}) = [H(M^\top \cdot)X(\cdot)]_{\downarrow \mathsf{M}}(\mathsf{f}) &= \frac{1}{|\det(\mathsf{M})|} \sum_{\mathsf{k}\in[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]} H(\mathsf{M}^\top(\mathsf{M}^{-\top}(\mathsf{f}-\mathsf{k})))X(\mathsf{M}^{-\top}(\mathsf{f}-\mathsf{k})) \\ &= \frac{1}{|\det(\mathsf{M})|} \sum_{\mathsf{k}\in[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]} H(\mathsf{f}-\mathsf{k})X(\mathsf{M}^{-\top}(\mathsf{f}-\mathsf{k})) \\ &= H(\mathsf{f})\frac{1}{|\det(\mathsf{M})|} \sum_{\mathsf{k}\in[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]} X(\mathsf{M}^{-\top}(\mathsf{f}-\mathsf{k})) \\ &= H(\mathsf{f})X_{\downarrow \mathsf{M}}(\mathsf{f}). \end{aligned}$$

Notons que cette dernière équivalence n'est valable que si $H(\mathsf{f})$ s'exprime sous la forme $H(\mathsf{f}) = \sum_{\mathsf{n}\in\mathbb{Z}^d} h[\mathsf{n}]e^{-2i\pi \mathsf{f}^\top \mathsf{n}}$. Ceci est vérifié si la réponse impulsionnelle de H est discrète, ce que nous supposerons par la suite.

1.3.2.2 Transformation polyphase

Supposons que nous avons un filtre H donné par sa réponse fréquentielle :

$$H(\mathsf{f}) = \sum_{\mathsf{n}\in\mathbb{Z}^d} h[\mathsf{n}]e^{-2i\pi \mathsf{f}^\top \mathsf{n}}.$$

Il est possible de décomposer cette réponse en considérant le partitionnement de $\mathbb{Z}^d$ par la matrice de sous-échantillonnage M :

$$\begin{aligned} H(\mathsf{f}) &= \sum_{\mathsf{k}\in[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]} \sum_{\mathsf{n}\in\mathbb{Z}^d} h[\mathsf{Mn}+\mathsf{k}]e^{-2i\pi \mathsf{f}^\top(\mathsf{Mn}+\mathsf{k})} \\ &= \sum_{\mathsf{k}\in[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]} e^{-2i\pi \mathsf{f}^\top \mathsf{k}} \sum_{\mathsf{n}\in\mathbb{Z}^d} h[\mathsf{Mn}+\mathsf{k}]e^{-2i\pi \mathsf{f}^\top \mathsf{Mn}}. \end{aligned}$$

Ainsi on définit les *composantes polyphases* $(\tilde{H}^{\mathsf{k}})_{\mathsf{k}\in[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]}$ de H de la manière suivante :

$$\tilde{H}^{\mathsf{k}}(\mathsf{f}) = \sum_{\mathsf{n}\in\mathbb{Z}^d} h[\mathsf{M}\mathsf{n}+\mathsf{k}]e^{-2i\pi \mathsf{f}^\top \mathsf{n}},$$

et H peut s'écrire en *représentation polyphase* :

$$H(\mathsf{f}) = \sum_{\mathsf{k}\in[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]} e^{-2i\pi \mathsf{f}^\top \mathsf{k}} \tilde{H}^{\mathsf{k}}(M^\top \mathsf{f}).$$

On remarque de plus que la réponse impulsionnelle des composantes polyphases se déduit aisément de la réponse impulsionnelle du filtre d'origine :

$$\tilde{h}^{\mathsf{k}}[\mathsf{t}] = h[\mathsf{M}\mathsf{t}+\mathsf{k}].$$

Ainsi le filtrage par H suivi du sous-échantillonage par M est effectué de manière équivalente en commençant par séparer le signal sur des lattices complémentaires puis en appliquant un filtrage par les composantes polyphases (Fig. 1.10).

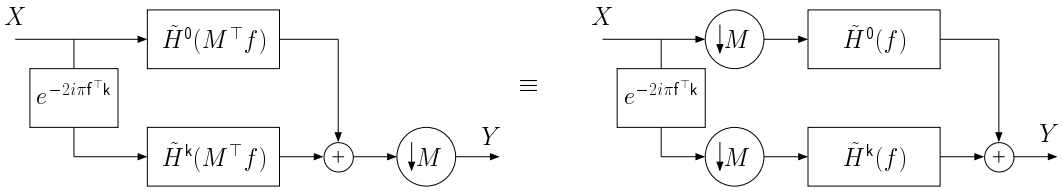

FIG. 1.10: Identité polyphase. Exemple pour une décomposition en deux canaux.

Ainsi, pour un banc de filtres à $|\det(M)|$ canaux, on obtient les réponses en sortie des filtres $(H_l)_{\mathsf{l}\in[\![0,|[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]|-1]\!]}$ par la relation suivante :

$$\mathsf{Y} = \tilde{\mathsf{H}}^\top \tilde{\mathsf{X}},$$

où la matrice $\tilde{\mathsf{H}}$ est appelée *matrice polyphase* du banc de filtres et est définie par [10] :

$$\tilde{\mathsf{H}}_{k,l} = \tilde{H}_l^{\mathsf{k}},$$

et $\tilde{\mathsf{X}}$ est le vecteur formée des composantes polyphases du signal X. On dit que (H_l) forme un n-uplet de filtres complémentaires lorsque $\det(\tilde{\mathsf{H}}) = 1$.

Le filtrage complet s'effectue donc en séparant le signal d'entrée en classes d'équivalences, puis en appliquant globalement la matrice polyphase du banc de filtres (Fig. 1.11). La synthèse s'effectue de manière similaire en multipliant les différents canaux par la matrice polyphase du banc de filtres de synthèse puis en recombinant les différentes sorties.

La condition de reconstruction parfaite décrite par l'équation 1.3 se traduit dans l'espace polyphase par la relation suivante :

$$\tilde{\mathsf{H}}(f)\tilde{\mathsf{G}}^\top(-f) = \mathsf{I} \tag{1.6}$$

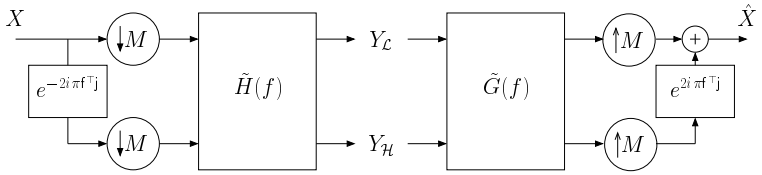

FIG. 1.11: Transformation polyphase d'un banc de filtres à deux canaux.

1.3.2.3 Factorisation en pas de lifting

Dans [10], Daubechies *et al* montrent qu'il est possible de factoriser la matrice polyphase d'un banc de filtres unidimensionnel à réponse impulsionnelle finie à deux canaux en un certain nombre d'étapes successives de prédiction (P_i) et de mise à jour (U_i), suivies d'une étape finale de mise à l'échelle (S). Cette décomposition, appelée *lifting*, a l'avantage de généraliser la notion d'ondelettes permettant la conception d'*ondelettes de seconde génération* sur des espaces où la notion de fréquence n'est plus définie. En outre, elle offre une mise en oeuvre efficace du filtrage et garantit la reconstruction parfaite (Fig. 1.12).

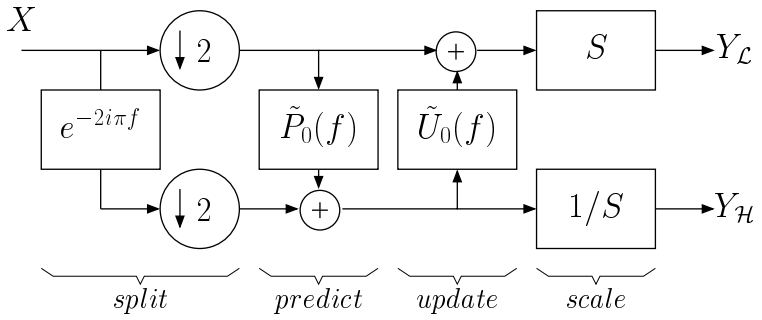

FIG. 1.12: Décomposition équivalente du banc de filtres d'analyse en étapes de lifting.

Les différentes étapes de lifting s'obtiennent par factorisation polynomiale. En effet, la transformée de Fourier discrète d'un filtre H constitue un polynôme en $e^{\pm 2i\pi f}$. Notons que ces filtres sont définis à un terme de déphasage $e^{2i\pi t}, t \in \mathbb{Z}$ près. On nomme de tels polynômes des *polynômes de Laurent*. Leur degré est obtenu par soustraction des puissances des monômes de plus haut degré et de plus petit degré. Cette valeur correspond à la longueur du filtre moins 1. La division Euclidienne, nécessaire à la factorisation, est définie sur ces polynômes. Cependant, elle n'est pas unique car il est possible de choisir arbitrairement le déphasage du reste de la division (ce qui peut modifier la valeur du quotient).

Les composantes polyphases $\tilde{H}^0$ et $\tilde{H}^1$ d'un filtre H sont également des polynômes de Laurent. Leur factorisation s'effectue en appliquant l'algorithme d'Euclide [11] qui trouve le polynôme de Laurent de degré le plus élevé divisant à la fois H^0 et H^1. Ce plus grand commun diviseur est défini à un terme de déphasage près, rendant cette

factorisation non unique. Cet algorithme se déroule de manière récursive en deux étapes.

Soit A_0, B_0 deux polynômes de Laurent, avec $\deg A_0 \geq \deg B_0$. L'algorithme d'Euclide consiste à appliquer successivement les étapes suivantes tant que B_i est non nul :

$$\begin{aligned} A_{i+1} &= B_i \\ B_{i+1} &= A_i - B_i Q_i \quad \text{(division Euclidienne de } A_i \text{ par } B_i\text{)} \end{aligned}$$

Notons que le degré de B décroît à chaque étape de l'algorithme, ce qui garantit la convergence. Dans le cas du banc de filtres à reconstruction parfaite, la paire de filtres d'analyse (H_0, H_1) est complémentaire, de même que la paire de filtres de synthèse (G_0, G_1). En effet, afin de satisfaire la propriété de reconstruction parfaite (Eq. 1.6), les déterminants des matrices polyphases H et G doivent être des monômes. En décalant et renormalisant H_1 par rapport à H_0, il est toujours possible d'obtenir $\det(\tilde{\mathsf{H}}) = 1$. De manière identique, on obtient $\det(\tilde{\mathsf{G}}) = 1$. Cette propriété implique que les composantes polyphases de ces filtres sont premières entre elles, car tout facteur commun diviserait aussi le déterminant de la matrice polyphase qui est égal à 1. On est donc assuré que le plus grand commun diviseur des composantes polyphases est un monôme. La factorisation n'étant pas unique il est possible de l'effectuer de telle sorte que ce monôme soit une constante. Pour tout filtre H du banc de filtres on obtient alors par application de l'algorithme d'Euclide sur ses composantes polyphases l'expression suivante :

$$\begin{pmatrix} S \\ 0 \end{pmatrix} = \prod_{i=N-1}^{0} \begin{pmatrix} 0 & 1 \\ 1 & -Q_i \end{pmatrix} \begin{pmatrix} H^0 \\ H^1 \end{pmatrix},$$

qui s'inverse aisément en la factorisation suivante :

$$\begin{pmatrix} H^0 \\ H^1 \end{pmatrix} = \prod_{i=0}^{N-1} \begin{pmatrix} Q_i & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} S \\ 0 \end{pmatrix}.$$

Les différentes étapes lifting de prédiction et de mise à jour de H sont donc obtenues à partir du quotient Q_i à chaque étape de l'algorithme d'Euclide. Ce quotient étant aussi un polynôme de Laurent, il correspond à un filtrage par un filtre à réponse impulsionnelle finie dont les coefficients sont donnés par ceux du polynôme. Le facteur d'échelle S est obtenu comme la valeur constante du plus grand commun diviseur des composantes polyphases de H.

Il est toujours possible d'obtenir une paire de filtres complémentaires telle que leur matrice polyphase $\tilde{\mathsf{H}}'$ se factorise sous la forme suivante :

$$\tilde{\mathsf{H}}' = \begin{pmatrix} H_0^0 & \hat{H}_1^0 \\ H_0^1 & \hat{H}_1^1 \end{pmatrix} = \prod_{i=0}^{N-1} \begin{pmatrix} Q_i & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} S & 0 \\ 0 & 1/S \end{pmatrix}.$$

En remarquant que

$$\begin{pmatrix} Q_i & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & Q_i \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ Q_i & 1 \end{pmatrix},$$

on obtient la factorisation suivante :

$$\tilde{\mathsf{H}}' = \prod_{i=0}^{\lfloor N/2 \rfloor} \begin{pmatrix} 1 & P_i \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ U_i & 1 \end{pmatrix} \begin{pmatrix} S & 0 \\ 0 & 1/S \end{pmatrix},$$

où $P_i = Q_{2i}$ et $U_i = Q_{2i+1}$. Or, $(H_0, \hat{H}_1)$ formant une paire de filtres complémentaires, il est possible d'obtenir H_1 en fonction de $\hat{H}_1$ et H_0 par la relation suivante :

$$H_1(f) = \hat{H}_1(f) + H_0(f)Q(2f).$$

La matrice polyphase $\tilde{\mathsf{H}}$ s'obtient alors à partir de $\tilde{\mathsf{H}}'$ en une dernière étape de lifting $P_{\lfloor N/2 \rfloor + 1} = S^2 Q$. En posant $M = \lfloor N/2 \rfloor + 1$, ceci conduit finalement à la factorisation lifting de $\tilde{\mathsf{H}}$:

$$\tilde{\mathsf{H}} = \prod_{i=0}^{M} \begin{pmatrix} 1 & P_i \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ U_i & 1 \end{pmatrix} \begin{pmatrix} S & 0 \\ 0 & 1/S \end{pmatrix},$$

où P_M et/ou U_M sont éventuellement nuls.

Cette factorisation n'est pas unique. En général, on préfère obtenir des pas liftings de degré 1 ou moins. Cela est possible en arrêtant la division Euclidienne des polynômes de Laurent dès que le degré du quotient atteint 1. Le reste de la division est toujours de degré inférieur au dividende, ce qui garantit encore la convergence de l'algorithme d'Euclide.

Exemple 6: Factorisation des filtres 9/7 biorthogonaux [12] définis par :

$$H_0 = a_0 + a_1(X^1 + X^{-1}) + a_2(X^2 + X^{-2}) + a_3(X^3 + X^{-3}) + a_4(X^4 + X^{-4})$$
$$H_1 = b_0 + b_1(X^1 + X^{-1}) + b_2(X^2 + X^{-2}) + b_3(X^3 + X^{-3}),$$

où les coefficients des polynômes sont donnés par :

	0	1	2	3	4
a	0.852699	0.377403	−0.110624	−0.0238495	0.0378285
b	0.788486	−0.418092	−0.0406894	0.0645389	

Les composantes polyphases de H_0 valent :

$$H_0^0 = a_0 + a_2(X^1 + X^{-1}) + a_4(X^2 + X^{-2})$$
$$H_0^1 = a_1(X^0 + X^{-1}) + a_3(X^1 + X^{-2}).$$

Le filtre étant symétrique, il est possible de trouver des étapes lifting qui le soient également. En se limitant de plus à des étapes minimales, on obtient des polynômes de

la forme $P_i = C(1+X)$ *et* $U_i = C(1+X^{-1})$, *où* C *est une constante. Nous allons donc chercher la valeur du premier pas lifting en résolvant :*

$$H_0^0 = H_0^1 \alpha(1+X) + R_0.$$

La solution s'obtient aisément en mettant en correspondance les termes en X^2 *puis en soustrayant :*

$$\begin{aligned}\alpha =& a_4/a_3 \\ R_0 =& H_0^0 - H_0^1\alpha(1+X) \\ =& (a_2 - a_4(\frac{a_1}{a_3}+1))(X+X^{-1}) + (a0 - 2\frac{a_1 a_4}{a_3}).\end{aligned}$$

Ce qui donne sous forme numérique $\alpha = -1.586134$ *et*

$$R_0 = 0.450159(X^1 + X^{-1}) + 2.049922.$$

De manière similaire, on cherche la valeur du deuxième pas lifting en résolvant :

$$H_0^1 = R_0\beta(1+X^{-1}) + R_1,$$

dont la solution est

$$\begin{aligned}\beta =& \frac{a_2 - a_4(\frac{a_1}{a_3}+1)}{a_3} = -0.052980 \\ R_1 =& H_0^1 - R_0\beta(1+X^{-1}) = 0.509858(1+X^{-1}).\end{aligned}$$

De même on résout $R_0 = R_1\gamma(1+X^{-1}) + R_2$ *pour obtenir* $\gamma = 0.882911$ *et*

$$R_2 = 1.149604.$$

Enfin la dernière étape s'obtient en résolvant $R_1 = R_2\gamma(1+X) + R_3$ *pour obtenir* $\delta = 0.443507$ *et* $R_3 = 0$. *L'algorithme d'Euclide s'arrête donc avec pour valeur du plus grand commun diviseur* $R_2 = 1.149604$. *Les étapes de lifting de ce filtre sont donc les suivantes :*

$$\begin{aligned}P_0 &= -1.586134(1+X) \\ U_0 &= -0.052980(1+X^{-1}) \\ P_1 &= 0.882911(1+X) \\ U_1 &= 0.443507(1+X^{-1}) \\ S &= 1.149604.\end{aligned}$$

On remarque que la factorisation du filtre H_1 *conduit aux mêmes étapes lifting hormis la dernière étape de mise à jour* U_1. *Il s'agit donc bien d'un filtre complémentaire à* H_0.

1.3.2.4 Mise en oeuvre du lifting

Ce type de factorisation est valable pour tous les filtres symétriques à réponse impulsionnelle finie, auxquels nous nous restreignons par la suite. Le filtrage s'implémente alors efficacement en une multiplication et une addition par étape lifting et par point de la classe d'équivalence correspondant à cette étape (Fig. 1.13). En effet, chaque point de la classe d'équivalence considérée est additionné à la somme de ses voisins pondérée par le facteur de l'étape lifting. Ceci permet de gagner jusqu'à un facteur 2 en nombre d'opérations par rapport à un filtrage direct. De plus, le résultat de cette opération peut être stocké à l'emplacement initial du point lifté, permettant une implémentation qui ne nécessite aucune mémoire supplémentaire. Ces deux raisons pratiques expliquent le succès que connaît l'implémentation lifting pour la réalisation de bancs de filtres.

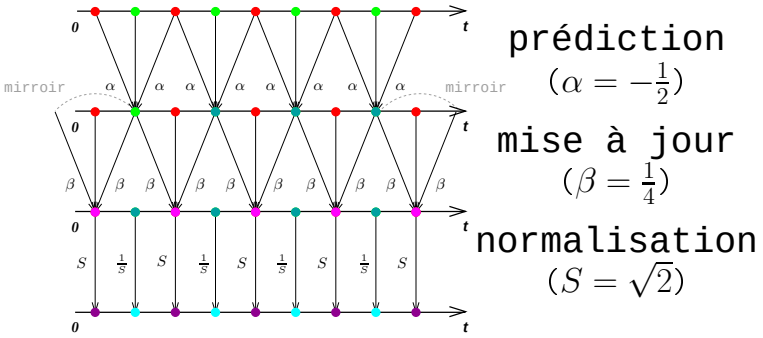

FIG. 1.13: Implémentation lifting d'un banc de filtres 5/3.

L'analyse s'effectue donc de la manière suivante. La grille d'échantillonnage est tout d'abord séparée en deux cosets :

$$C : \begin{cases} y_0[t] \leftarrow x[2t], \\ y_1[t] \leftarrow x[2t+1]. \end{cases}$$

L'étape de prédiction consiste à prédire chaque échantillon de y_1 à partir de ses voisins dans y_0 par la relation suivante :

$$P(\alpha_k) : y_1[t] \leftarrow y_1[t] + \alpha_k(y_0[t] + y_0[t+1]). \tag{1.7}$$

De même, l'étape de mise à jour consiste à ajuster la valeur de chaque échantillon de y_0 à partir de ses voisins dans y_1 par la relation :

$$U(\beta_k) : y_0[t] \leftarrow y_0[t] + \beta_k(y_1[t] + y_1[t-1]). \tag{1.8}$$

Ces étapes sont réalisées l'une après l'autre alternativement. Une étape finale de mise à l'échelle ajuste l'amplitude des coefficients obtenus :

$$S(\chi) : \begin{cases} y_0[t] \leftarrow \chi y_0[t], \\ y_1[t] \leftarrow \frac{1}{\chi} y_1[t]. \end{cases} \tag{1.9}$$

On obtient alors, respectivement, l'approximation de x dans y_0 correspondant au filtrage par H_0 suivi d'une décimation, et l'erreur de prédiction dans y_1 correspondant au filtrage par H_1 suivi d'une décimation. La synthèse du signal reconstruit $\hat{x}$ s'effectue en inversant les étapes de lifting, c'est à dire en appliquant successivement

$$S(\frac{1}{\chi}), U(-\alpha_M), P(-\beta_M), ..., U(-\alpha_0), P(-\beta_0)$$

puis en regroupant y_0 et y_1 :

$$C^{-1} : \begin{cases} \hat{x}[2t] \leftarrow y_0[t], \\ \hat{x}[2t+1] \leftarrow y_1[t]. \end{cases}$$

1.4 Théorie de l'information et compression

Afin d'aborder la compression de manière rigoureuse, nous rappelons ici certains résultats fondamentaux de théorie de l'information utilisés en compression. Un développement plus détaillé, ainsi que les preuves de la plupart des résultats présentés se trouvent dans [13]. Cette théorie, introduite par Claude Shannon en 1948 [14], établit les limites atteignables pour la communication et la compression de données numériques. En particulier, il est prouvé que les réalisations d'un processus aléatoire discret dont on connaît la statistique (une source) ne peuvent être représentées sans erreur en un nombre moyen de bits inférieur à une constante appelée *entropie* de la source (théorème de codage source). Il est également démontré qu'un message peut être transmis à un taux d'erreur arbitrairement faible, tant que son débit n'excède pas la capacité du canal dont on connaît la statistique (théorème de codage canal). Notons cependant que ces résultats sont des valeurs asymptotiques sous les hypothèses de longueur de message infinie, de capacité de calcul infinie et de connaissance parfaite des statistiques.

Nous nous intéressons dans cette thèse principalement au codage de source plutôt qu'au codage de canal, bien qu'il existe une dualité forte entre ces deux problèmes. En particulier, nous présentons les différentes mesures d'information ainsi que la limite de compression sans perte d'une source discrète. La théorie débit-distorsion et son lien avec la quantification de sources continues sont détaillés. Certains outils pratiques de compression sont aussi présentés.

1.4.1 Mesures d'information et compression sans perte

Pour pouvoir parler de théorie de l'information, il est tout d'abord nécessaire de définir ce que l'on entend par information et comment la mesurer. Soit $\mathcal{A}$ l'ensemble des *symboles* formant les messages que l'on désire écrire. Un message est formalisé comme un processus $\mathcal{X} = (X_t)_{t \in \mathbb{N}}$, où chaque variable aléatoire X_t est définie sur l'*alphabet* $\mathcal{A}$.

La probabilité que le message soit constitué d'un symbole particulier x_t à un instant t est décrite par $\mathbb{P}(X_t = x_t)$.

La quantité d'information apportée par la lecture du symbole x_t à l'instant t se définit alors comme la suppression de l'incertitude sur X_t apportée par la connaissance de cette réalisation particulière x_t. Elle est définie par :

$$\mathbb{I}(X_t = x_t) = -\log_2(\mathbb{P}(X_t = x_t))$$

et s'exprime en bits. Elle correspond en effet au nombre moyen de questions/réponses binaires (oui/non) nécessaires pour en déduire de manière certaine que la réalisation particulière de X_t est x_t lorsque l'on cherche à déterminer X_t en un nombre de questions minimal. On appelle aussi la quantité $\mathbb{I}(X_t = x_t)$ le *coût de codage* de x_t.

1.4.1.1 Entropie

Le débit entropique d'une source $\mathcal{X}$ est une mesure de la quantité d'information moyenne par symbole à transmettre ou à stocker pour représenter de façon univoque toute réalisation possible de cette source.

Pour cela commençons par la définition de l'entropie d'une variable aléatoire discrète X. Considérons la source décrite par $X_t = X$ pour tout instant. La quantité d'information à transmettre à chaque instant vaut $\mathbb{I}(X)$, qui est également une variable aléatoire. La quantité d'information moyenne par symbole de cette source est alors donnée par l'espérance de $\mathbb{I}(X)$. Cette quantité est appelée l'entropie de X,

$$H(X) = -\sum_{x \in \mathcal{A}} \mathbb{P}(X = x) \log_2(\mathbb{P}(X = x)),$$

exprimée en bits/symbole. L'entropie est une mesure toujours positive. Elle est maximale lorsque X suit une loi uniforme, et vaut alors $\log_2(|\mathcal{A}|)$. Elle est nulle lorsque X est connue de manière certaine, c'est à dire quand X est déterministe.

Passons à présent au cas de deux variables aléatoires discrètes X et Y. En considérant la variable aléatoire discrète formée par le couple (X, Y) et en utilisant la définition précédente on obtient l'*entropie jointe* de X, Y :

$$H(X, Y) = -\sum_{x,y \in \mathcal{A}} \mathbb{P}(X = x, Y = y) \log_2(\mathbb{P}(X = x, Y = y)).$$

L'entropie jointe vérifie la propriété suivante :

$$H(X, Y) \leq H(X) + H(Y)$$

avec égalité lorsque les variables sont indépendantes.

On définit également l'*entropie conditionnelle* de X sachant Y par :

$$H(X|Y) = -\sum_{x,y \in \mathcal{A}} \mathbb{P}(X = x, Y = y) \log_2(\mathbb{P}(X = x|Y = y)).$$

qui correspond à l'espérance du coût de codage de X moyennant la connaissance de Y. On a alors la propriété suivante :

$$H(X,Y) = H(X) + H(Y|X) = H(Y) + H(X|Y).$$

On se réfère parfois à $H(X)$ comme à l'*entropie absolue* ou de *premier ordre*, tandis que $H(X|Y)$ est également appelée entropie de *second ordre*.

En généralisant la notion d'entropie jointe, on obtient la définition du débit entropique d'une source $\mathcal{X}$:

$$H(\mathcal{X}) = \lim_{N\to\infty} \frac{1}{N} H(X_0, ..., X_{N-1})$$

lorsque la limite existe.

La compression sans perte consiste à représenter un message en un nombre moyen minimum de bits. Pour cela, on associe à chaque réalisation possible de la source une séquence de bits de longueur variable la représentant de manière univoque. L'espérance de la longueur de cette séquence est au minimum égale à l'entropie de la source multipliée par le nombre de symboles du message. En d'autres termes, le débit entropique de $\mathcal{X}$ constitue la limite théorique de compression atteignable pour cette source, exprimée en nombre de bits moyen par symbole.

En règle générale, on ne connaît malheureusement pas totalement la statistique exacte de la source $\mathcal{X}$. De plus, l'estimation de la loi de la source est souvent problématique. En effet, la quantité de réalisations de la source à observer pour obtenir une loi statistiquement fiable est souvent trop importante.

Ainsi, on suppose généralement que $\mathcal{X}$ est un processus markovien, dont la dépendance statistique se restreint à un voisinage limité. Cette approximation entraîne nécessairement la considération d'une source markovienne d'entropie supérieure à la source réelle. Ainsi, on est souvent amené à réaliser un compromis entre fiabilité de la statistique estimée et perte en performance de compression. En poussant cette modélisation à l'extrême, on suppose parfois que $\mathcal{X}$ est formé d'échantillons indépendants.

On voit alors l'intérêt de traiter le signal pour en réduire la dépendance statistique avant d'effectuer la compression proprement dite. Ceci permet d'utiliser des modèles markoviens d'ordres faibles tout en représentant correctement la statistique de la source. Ce traitement est en général effectué par la transformée, dont le but est de décorréler la source $\mathcal{X}$ pour obtenir une source $\mathcal{Y}$ de coefficients moins dépendante qui sera compressée.

1.4.1.2 Entropie relative et information mutuelle

Se pose à présent la question de définir ce qui constitue une bonne approximation $\mathcal{Y}$ de la source $\mathcal{X}$. Pour cela on définit l'*entropie relative* entre deux variables aléatoires X et Y par :

$$D(X||Y) = \sum_{x\in\mathcal{A}} \mathbb{P}(X=x) \log_2 \left(\frac{\mathbb{P}(X=x)}{\mathbb{P}(Y=x)} \right).$$

Elle mesure le surcoût de codage induit par le choix du modèle statistique $\mathbb{P}(Y)$ pour compresser la variable X. On appelle également cette quantité la *divergence de Kullback-Leibler* entre $\mathbb{P}(X)$ et $\mathbb{P}(Y)$. L'entropie relative est toujours positive et s'annule uniquement si $X = Y$. Notons que cette mesure n'est pas symétrique et ne constitue donc pas une distance.

L'*information mutuelle* entre X et Y mesure la quantité d'information contenue par l'une des variables sur l'autre variable. Elle se définit comme :

$$I(X;Y) = \sum_{x,y \in \mathcal{A}} \mathbb{P}(X = x, Y = y) \log_2 \left(\frac{\mathbb{P}(X = x, Y = y)}{\mathbb{P}(Y = y)\mathbb{P}(X = x)} \right).$$

Ceci correspond à la divergence de Kullback-Leibler entre la loi jointe $\mathbb{P}(X, Y)$ et la loi produit $\mathbb{P}(X)\mathbb{P}(Y)$, qui revient à supposer les variables X et Y indépendantes. Ainsi, plus l'information mutuelle entre deux variables est faible, moins il sera coûteux de les supposer indépendantes, et ce surcoût est donné par l'information mutuelle.

L'information mutuelle s'exprime également comme la diminution d'entropie sur X introduite par la connaissance de Y en remarquant que :

$$I(X;Y) = H(X) - H(X|Y).$$

Elle est de plus symétrique, et on a $I(X;X) = H(X)$, d'où le terme d'*information propre* utilisé parfois pour désigner $H(X)$.

Ainsi, étant donné un modèle Y vectoriel de la source $\mathcal{X}$, on peut mesurer l'impact sur le coût de codage de l'utilisation d'un autre modèle Y' plus simple (tel que la longueur du vecteur Y' est inférieure à celle de Y) en calculant l'information mutuelle $I(Y;Y')$ entre ces deux modèles.

1.4.1.3 Codeur arithmétique

Le codeur arithmétique [15] [16] est une technique de codage permettant d'atteindre le débit entropique d'une source au bit près en un temps de calcul linéaire par rapport à la longueur de la séquence à coder. Son principe trouve ses origines dans le papier original de Shannon [14] et le code d'Elias [17], qui proposent d'associer une longueur de code à chaque séquence de $\mathcal{X}$ en fonction de sa probabilité d'apparition.

Le codeur arithmétique original fonctionne de la manière suivante (Fig. 1.14). Soit I_0 l'intervalle $[0, 1[$ et X une séquence de longueur $L(\mathsf{X})$ issue de la source $\mathcal{X}$. Chaque symbole X_t est codé successivement en séparant l'intervalle I_t en $|\mathcal{A}|$ sous-intervalles $(I_t^i)_{i \in [\![0, |\mathcal{A}|-1]\!]}$. Chaque sous-intervalle I_t^i a une longueur proportionnelle à la probabilité d'obtenir le i-ième symbole dans $\mathcal{A}$ à l'instant t sachant l'ensemble des réalisations précédentes. L'intervalle I_{t+1} est choisi comme le sous-intervalle correspondant à la réalisation de X_t. Une fois toute la séquence $X_0 ... X_{L(\mathsf{X})-1}$ codée, on obtient un intervalle final $I_{L(\mathsf{X})}$ de longueur :

$$|I_{L(\mathsf{X})}| = \prod_{t=0}^{L-1} \mathbb{P}(X_t = x_t | X_0 ... X_{t-1} = x_0 ... x_{t-1}) = \mathbb{P}(\mathsf{X} = \mathsf{x}).$$

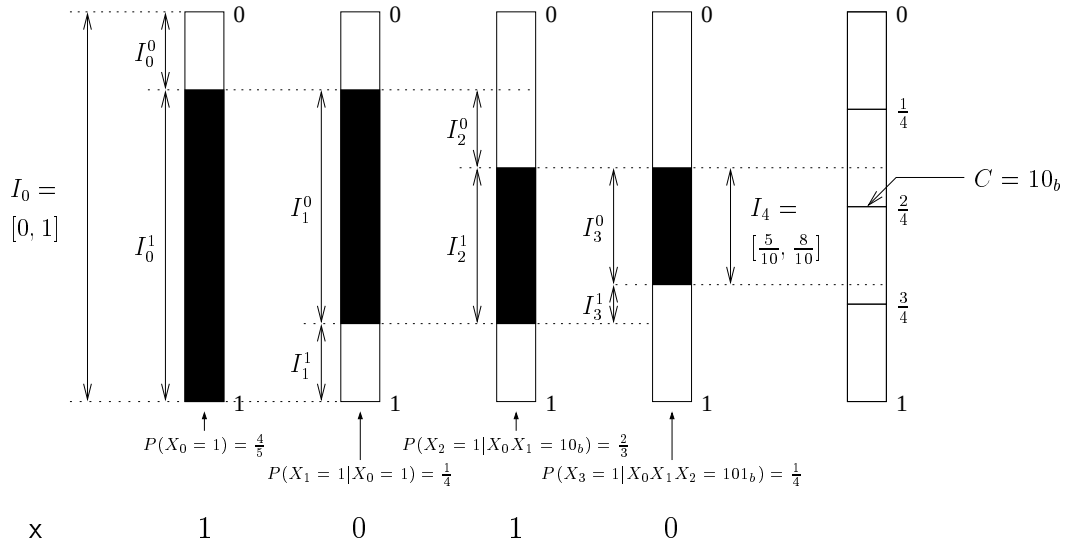

FIG. 1.14: Codeur arithmétique. L'intervalle $[0,1]$ est subdivisé successivement en fonction de la probabilité d'apparition de chaque symbole. Le mot de code C représentant la séquence est le numérateur d'une fraction appartenant à l'intervalle final. Le décodage s'effectue en testant l'appartenance de ce point aux intervalles des symboles.

Notons $L(C) = \lceil -\log_2(|I_{L(\mathsf{X})}|)\rceil$. On est assuré de trouver un point dans l'intervalle $I_{L(\mathsf{X})}$ représentable sous forme de fraction rationnelle $\frac{C}{2^{L(C)}}$, où $C \in [\![0, 2^{L(C)} - 1]\!]$. La représentation binaire de C donne le code de longueur $L(C)$ utilisé pour représenter la séquence X. Le décodage s'effectue de manière similaire, en testant à chaque instant t à quel intervalle appartient la fraction $\frac{C}{2^{L(C)}}$ et en émettant le symbole correspondant. L'espérance de longueur de ce code est donnée par :

$$\begin{aligned}\mathbb{E}\left[L(C)\right] &= \sum_{\mathsf{x}\in\mathcal{A}^L} \mathbb{P}(\mathsf{X}=\mathsf{x})\lceil -\log_2(|I_{L(\mathsf{X})}|)\rceil \\ &\leq \sum_{\mathsf{x}\in\mathcal{A}^L} \mathbb{P}(\mathsf{X}=\mathsf{x})(-\log_2(P(\mathsf{X}=\mathsf{x}))+1) = H(\mathsf{X})+1\end{aligned}$$

Ce code est donc très performant d'où son intérêt pratique. En général, on se restreint au codeur arithmétique binaire car toute loi sur une source discrète peut être vue comme une loi d'ordre supérieur sur une source binaire. La transformation d'une source discrète en source binaire est appelée *binarisation*. Un autre intérêt majeur du codeur arithmétique est de pouvoir (en général) commencer à envoyer le mot de code C avant d'obtenir la fin de la séquence X. Ceci se réalise en effectuant des renormalisations successives de l'intervalle de codage (Fig. 1.15). De même, on commence à décoder des symboles avant d'avoir reçu l'ensemble du code C. Un tel codeur requiert en théorie une précision infinie sur les intervalles de codage. En pratique, on se limite à des positions et des tailles d'intervalle finies. Ceci entraîne une légère sous-optimalité de ces codeurs

quasi-arithmétiques [18] car l'espace des probabilités représentables est également fini. Ces codeurs sont utilisés dans les systèmes de compression d'image actuels, comme par exemple dans le codeur entropique EBCOT [19] proposé par la norme JPEG2000.

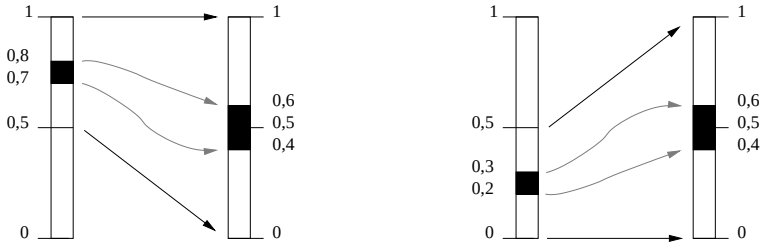

FIG. 1.15: Renormalisation de l'intervalle du codeur arithmétique. Le mot de code C commence par 0 ; ce bit est envoyé et l'intervalle $[0, \frac{1}{2}]$ est renormalisé en $[0, 1]$ [gauche]. Renormalisation lorsque le mot de code C commence par 1 [droite].

Les probabilités de la source sont usuellement estimées à la volée, de manière adaptative, lors du codage. Un *contexte*, formé d'un sous ensemble de symboles préalablement codés, conditionne la loi utilisée pour coder le symbole courant. Ceci correspond à une approximation markovienne de la source. Plus le nombre de voisins pris en compte est important, plus l'ordre du modèle markovien est élevé. Le CABAC [20], utilisé dans la norme vidéo H.264, est un exemple d'un tel codeur.

1.4.2 Théorie débit-distorsion et quantification

Les développements précédents se placent dans le cadre de sources discrètes, dont les réalisations sont décrites parfaitement pour peu qu'on les représente en un nombre de bits moyens supérieur à l'entropie. Les signaux réels sont cependant souvent des variables aléatoires continues. Pour décrire toute réalisation parfaitement il faudrait malheureusement un nombre infini de bits. On s'autorise alors à représenter les réalisations de la source continue de manière imparfaite. Pour cela, un nombre fini de séquences représentatives est choisi, de manière à pouvoir coder en un nombre fini de bits toute réalisation de la source, moyennant une certaine *distorsion*. Le nombre et la distribution de ces séquences influence le *débit* entropique nécessaire pour les coder. On est en général amené à réaliser un compromis entre le débit de codage et la distorsion introduite sur la source. La théorie débit-distorsion établit les bornes atteignables en terme de débit pour une distorsion donnée et réciproquement, dans le cas général ainsi que pour diverses sources théoriques. Le passage d'une source continue $\mathcal{X}$ à la source discrète $\mathcal{X}_Q$ des représentants est appelé *quantification*.

La fonction débit-distorsion, notée $D(R)$, se définit alors comme la distorsion minimale introduite sur la source $\mathcal{X}$ sous contrainte de débit entropique R pour la source $\mathcal{X}_Q$. On ne sait pas en général déterminer cette fonction de manière analytique. L'algorithme de Blahut [21] permet cependant de l'approcher de manière numérique. La recherche du quantificateur permettant d'atteindre cette limite peut être vue comme un

problème d'optimisation sous contrainte, qu'on résout généralement par des approches lagrangiennes.

La mesure de distorsion la plus couramment utilisée est l'erreur quadratique moyenne (MSE). Dans le cadre des signaux visuels l'utilisation d'une mesure psychovisuelle serait plus adaptée. Bien que l'utilisation de la MSE comme mesure de la distorsion visuelle soit critiquable, aucune autre mesure psychovisuelle ne semble faire l'objet d'un consensus dans la communauté de la compression d'images. On utilisera donc également la MSE comme mesure de distorsion dans cette thèse.

Un résumé des nombreuses techniques de quantification actuelles est disponible dans [22]. Nous ne nous intéressons dans cette thèse qu'aux techniques utilisées dans les systèmes de compression d'images, dont principalement la quantification scalaire et la quantification TCQ. Cette dernière sera par ailleurs présentée de manière détaillée au chapitre 5.

1.4.2.1 Quantification scalaire

La quantification scalaire consiste à représenter toute réalisation d'une variable aléatoire continue X par une réalisation de la variable aléatoire discrète X_Q définie sur un alphabet $\mathcal{A}$. Pour cela on associe à x la valeur quantifiée $x_q = Q(x)$, où Q est une fonction de quantification (non-linéaire) de $\mathbb{R}$ dans $\mathbb{Z}$. La fonction de déquantification notée $Q^\star$ consiste à associer une valeur de reconstruction réelle $\hat{x}$ à chaque valeur quantifiée, i.e. $\hat{x} = Q^\star[x_q]$. On cherche alors à minimiser l'espérance de l'erreur de quantification :

$$\mathbb{E}\left[(\hat{X} - X)^2\right].$$

L'algorithme Lloyd-Max [23] [24] donne la solution optimale de ce problème de manière itérative pour une taille d'alphabet $|\mathcal{A}|$ donnée. Cet algorithme part d'un dictionnaire de valeurs de reconstructions $\mathcal{D} = Q^\star[\mathcal{A}]$ quelconque (souvent uniforme). Le quantificateur s'en déduit en posant $Q(x) = \underset{x_q \in \mathcal{A}}{\operatorname{argmin}} \left((Q^\star[x_q] - x)^2\right)$, qui consiste à trouver le plus proche point de x dans $\mathcal{D}$. En itérant les deux étapes suivantes on s'assure que l'espérance de distorsion diminue à chaque itération :

$$\forall i \in [\![1, |\mathcal{A}| - 1]\!],\ t_i \leftarrow \frac{1}{2}(\hat{x}_i + \hat{x}_{i+1})$$

$$\forall i \in [\![0, |\mathcal{A}| - 1]\!],\ \hat{x}_i \leftarrow \frac{\int_{t_i}^{t_{i+1}} x\mathbb{P}(X = x)dx}{\int_{t_i}^{t_{i+1}} \mathbb{P}(X = x)dx}$$

où $\hat{x}_0, ..., \hat{x}_{|\mathcal{A}|-1}$ représente la séquence ordonnée des éléments de $\mathcal{D}$ et $\begin{cases} t_0 & = -\infty \\ t_{|\mathcal{A}|} & = +\infty \end{cases}$.

On arrête en général l'algorithme lorsque la variation relative de distorsion est suffisamment faible. On obtient alors le quantificateur optimal pour la source X dans le

sens où il minimise l'erreur de reconstruction si l'on transmet directement x_q. Notons cependant que la loi de X_Q n'est en général pas uniforme et qu'il est donc encore possible de compresser la source quantifiée.

Lorsque cette quantification est suivie d'un codage entropique, l'algorithme Lloyd-Max ne fournit donc généralement pas le dictionnaire optimal de quantification. Il est cependant possible de construire un algorithme de Lloyd-Max *contraint en entropie* tenant compte du débit en sortie du codeur entropique pour obtenir le dictionnaire optimal au débit de codage de la source désiré. Cette méthode implique toutefois de transmettre le dictionnaire ou les paramètres de la source au décodeur, dont le débit moyen de codage est fixé une fois pour toutes. Enfin, il a été démontré dans [25] que les dictionnaires de quantification contrainte en entropie tendent à haut débit vers des lattices de centroïdes espacés régulièrement.

Ainsi, il est parfois avantageux de se contenter d'un quantificateur uniforme, permettant également une quantification *emboîtée* de la source. On considère alors un ensemble de quantificateurs de pas $\Delta_i = 2^{-i}\Delta_0$, où Δ_0 est l'amplitude maximale du signal. Ces quantificateurs codent la source à plusieurs débits dans le même flux binaire, offrant ainsi une représentation progressive de celle-ci. En effet, la réalisation quantifiée au pas Δ_{i+1} s'obtient simplement de la réalisation quantifiée au pas Δ_i en fournissant le bit de poids faible, codé entropiquement. Notons que ce raisonnement se généralise au cas vectoriel lorsque les dictionnaires forment des lattices, en considérant leurs sous-lattices et un système de nombre associé [26].

Les quantificateurs scalaires utilisés en pratique dans les systèmes de compression d'image sont des quantificateurs *à zone morte* dans lesquels l'intervalle de quantification centré en l'origine est de taille multiple de la taille des autres intervalles de quantification. Ces quantificateurs sont plus adaptés à la quantification de sources dont la distribution est centrée en zéro et de forte densité en son voisinage. Généralement, la taille de la zone morte est double de celle des autres zones, ce qui correspond au quantificateur à zone morte optimal pour les sources laplaciennes à bas débit. Les réalisations de faible amplitude sont alors reconstruites en zéro et traitées de manière particulière pour les coder efficacement. Ce type de quantificateurs reste également emboîtable.

1.4.2.2 Quantification vectorielle

Outre le gain apporté par rapport à la quantification scalaire uniforme en adaptant le dictionnaire à la distribution de la source, il est possible d'obtenir un gain structurel en considérant plusieurs symboles de la source à la fois. En effet, une meilleure répartition des centroïdes dans l'espace permet de réduire la distorsion introduite. On considère alors la quantification de vecteurs formés de plusieurs réalisations consécutives de la source.

L'algorithme Lloyd-Max présenté ci-dessus, ainsi que sa version contrainte en entropie, se généralisent en dimension finie [27] [28]. Cependant sa convergence dépend fortement de l'initialisation et le temps de calcul du dictionnaire optimal varie de manière exponentielle avec la dimension des vecteurs considérés. C'est pourquoi il existe de nombreuses techniques de quantification vectorielle sous-optimales atteignant de

bonnes performances tout en ayant un coût de calcul limité.

A haut débit, la performance débit-distorsion d'un quantificateur vectoriel est bornée par :

$$D_d(R) = \frac{(\Gamma(1+\frac{d}{2}))^{\frac{2}{d}}e}{(1+\frac{d}{2})} D(R),$$

où $D(R)$ est la fonction débit-distorsion liée à la quantification scalaire de la source (c'est à dire en considérant chaque symbole X_t indépendamment, ce qui correspond à une fonction Q séparable). En faisant tendre d vers l'infini, on constate qu'il est possible d'obtenir un gain structurel asymptotique de 1.53 dB en rapport signal sur bruit (SNR) par rapport au quantificateur scalaire optimal. Ce résultat peut paraître surprenant dans le sens où, même pour une source $\mathcal{X}$ formée de symboles indépendants, le quantificateur scalaire n'est pas optimal.

Exemple 7: Considérons une source de symboles X indépendants et identiquement distribués selon la loi uniforme définie sur l'intervalle $[0,1]$. Le quantificateur scalaire optimal pour cette source est le quantificateur uniforme dont le dictionnaire est formé de N valeurs régulièrement espacées d'un pas $\Delta = \frac{1}{N}$ et équiprobables. La distorsion se calcule explicitement comme :

$$\mathbb{E}\left[(\hat{X}-X)^2\right] = \int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} t^2 \frac{1}{\Delta} dt = \frac{\Delta^2}{12}.$$

Comme chaque valeur quantifiée est équiprobable, l'entropie de la source quantifiée est donnée par $H(X_q) = \log_2(N)$. On obtient donc explicitement la fonction débit-distorsion pour la quantification scalaire d'une source uniforme :

$$D^U(R) = \frac{1}{12} 2^{-2R}.$$

La puissance du signal est donnée par la variance de X qui vaut $\frac{1}{12}$. On en déduit que le SNR est amélioré de $10\log_{10}(4) \simeq 6$ dB par bit par symbole utilisé pour décrire la source uniforme.

En comparaison, l'utilisation d'une quantification vectorielle optimale 2D permet de coder cette même source au même débit *en améliorant le SNR de 0.17 dB. En dimension 24, ce gain passe à 1.03dB en utilisant un quantificateur sur la lattice λ_{26} [29].*

Le problème de trouver un quantificateur optimal pour une source uniforme est lié de très près au problème d'empilement de sphères (*sphere packing*) en géométrie. Ce problème consiste à trouver comment placer N sphères de même rayon pour remplir au mieux un certain volume. En dimension 1 la réponse à ce problème mène au quantificateur uniforme. En dimension 2 on obtient un quantificateur dont le dictionnaire forme une lattice hexagonale. Il a été prouvé très récemment que le meilleur empilement en dimension 3 est également une lattice. En revanche, on ne connaît pour l'instant pas la structure du quantificateur uniforme optimal dans le cas général. Nous nous intéressons

ici uniquement au cas des quantificateurs vectoriels sur lattices. En particulier, nous proposons une extension du quantificateur par treillis (TCQ) au chapitre 5. Ce quantificateur vectoriel est un cas particulier d'un quantificateur sur une lattice issue d'un code convolutif, dont les détails seront présentés dans ce même chapitre.

1.5 Optimisation débit-distorsion

La qualité du signal reconstruit et le débit nécessaire pour le représenter est toujours l'objet d'un compromis. L'optimisation débit-distorsion consiste à rechercher la configuration de codage optimale du signal x, définie par le vecteur de paramètres p, en fonction du débit ou de distorsion voulue. La limite est généralement placée sur le débit R que l'on désire inférieur à un débit cible R_t. Le problème se formalise alors comme la minimisation de la distorsion D introduite sur x sous contrainte de débit $R \leq R_t$, à savoir :

$$\min_{\substack{\mathsf{p}\in\mathcal{P} \\ R(\mathsf{p})\leq R_t}} D(\mathsf{p})$$

où p parcours l'ensemble $\mathcal{P}$ des configurations de codage possibles.

Dans le cas où D et R sont des fonctions convexes de p, conditions généralement vérifiées en pratique pour le codage par transformées, ce problème contraint se résout en deux étapes par la méthode de Lagrange[1] en posant la fonction suivante :

$$J(\mathsf{p}, \lambda) = D(\mathsf{p}) + \lambda(R(\mathsf{p}) - R_t),$$

où λ est un coefficient réel positif. En cherchant les paramètres $\mathsf{p}^\star(\lambda)$ minimisant cette fonction J sur l'ensemble des configurations de codage possibles à λ fixé, puis en cherchant la valeur de λ pour laquelle $\mathsf{p}^\star(\lambda)$ correspond à la contrainte de débit $R(\mathsf{p}^\star(\lambda)) \leq R_t$, la configuration optimale satisfaisant le compromis débit-distorsion est trouvée. En d'autres termes, le problème est séparé en :

$$\min_{\lambda\in\mathbb{R}_+} \min_{\mathsf{p}\in\mathcal{P}} J(\mathsf{p}, \lambda)$$

Cette optimisation est généralement trop complexe à réaliser directement. L'optimisation globale des paramètres de codage du signal x est alors ramenée à l'optimisation locale de N sous parties x_i supposées indépendantes et formant une partition de x. Chacune de ses parties contribue au débit total et à la distorsion totale sur le signal x. Plus spécifiquement, l'espace des paramètres $\mathcal{P}$ est partitionné en $\bigcup_{i=1}^{N} \mathcal{P}_i$, le débit et la distorsion étant approximés par :

[1] plus précisement il s'agit de la méthode des multiplicateurs de Kuhn et Tucker mais la dénomination de multiplicateurs de Lagrange, correspondant à une contrainte d'égalité, est plus souvent employée dans les publications.

$$\begin{aligned} D(\mathsf{p}) &\approx \sum_{i=1}^{N} D_i(\mathsf{p}_i) \\ R(\mathsf{p}) &\approx \sum_{i=1}^{N} R_i(\mathsf{p}_i) \end{aligned}$$

Le but de la transformée dans les schémas de codage est de s'assurer que ces approximations sont bien valides. D'une part, l'hypothèse d'indépendance est relativement bien vérifiée. Ainsi, la somme des débits R_i obtenus après codage de chaque sous partie x_i du signal x est à peine supérieure au débit total R minimum atteignable donné par l'entropie de X. D'autre part, la propriété d'orthogonalité permet de supposer l'additivité des distorsions dans le domaine transformé. Il est alors possible de s'affranchir de la transformée inverse pour déterminer $D(\mathsf{p})$.

En pratique, l'ensemble des paramètres testés est fini pour que l'optimisation se termine en un temps de calcul raisonnable. Une manière efficace de la réaliser pour différentes contraintes de débit est d'estimer dans un premier temps les fonctions débit-distorsion $D_i(R_i)$ puis de déterminer le débit R_i optimal alloué à chaque sous partie du signal x. Une méthode possible pour déterminer $D_i(R_i)$ est de parcourir un certain nombre de points de l'espace des paramètres puis d'en calculer l'enveloppe convexe. Une autre possibilité est d'utiliser un modèle de X_i dont la courbe débit-distorsion et les paramètres permettant de l'atteindre sont connus (Fig. 1.16).

Une fois les fonctions débit-distorsion déterminées le problème d'allocation de débit se résout sous la forme suivante :

$$\min_{\lambda \in \mathbb{R}_+} \left(\left(\sum_{i=1}^{N} \min_{R_i \in \mathbb{R}} \left(D_i(R_i) + \lambda R_i \right) \right) - \lambda R_t \right).$$

Ce minimum est atteint en $(R_1^\star, \ldots, R_N^\star, \lambda^\star)$ lorsque toutes les dérivées correspondantes s'annulent, à savoir :

$$\begin{aligned} &\forall i \in [\![1, N]\!], \tfrac{\partial D_i}{\partial R_i}(R_i^\star) + \lambda^\star = 0 \\ &\text{et } \sum_{i=1}^{N} R_i^\star = R_t. \end{aligned}$$

Sous cette forme, il est aisé de voir que l'allocation de débit est réalisée en choisissant une valeur de λ correspondant à une pente commune sur les courbes débit-distorsion $D_i(R_i)$, puis en la faisant varier jusqu'à ce que la somme des débits sur chaque sous partie corresponde au débit total autorisé (Fig. 1.17). C'est la procédure qui est généralement adoptée dans les codeurs actuels, la recherche de pente étant souvent réalisée par dichotomie.

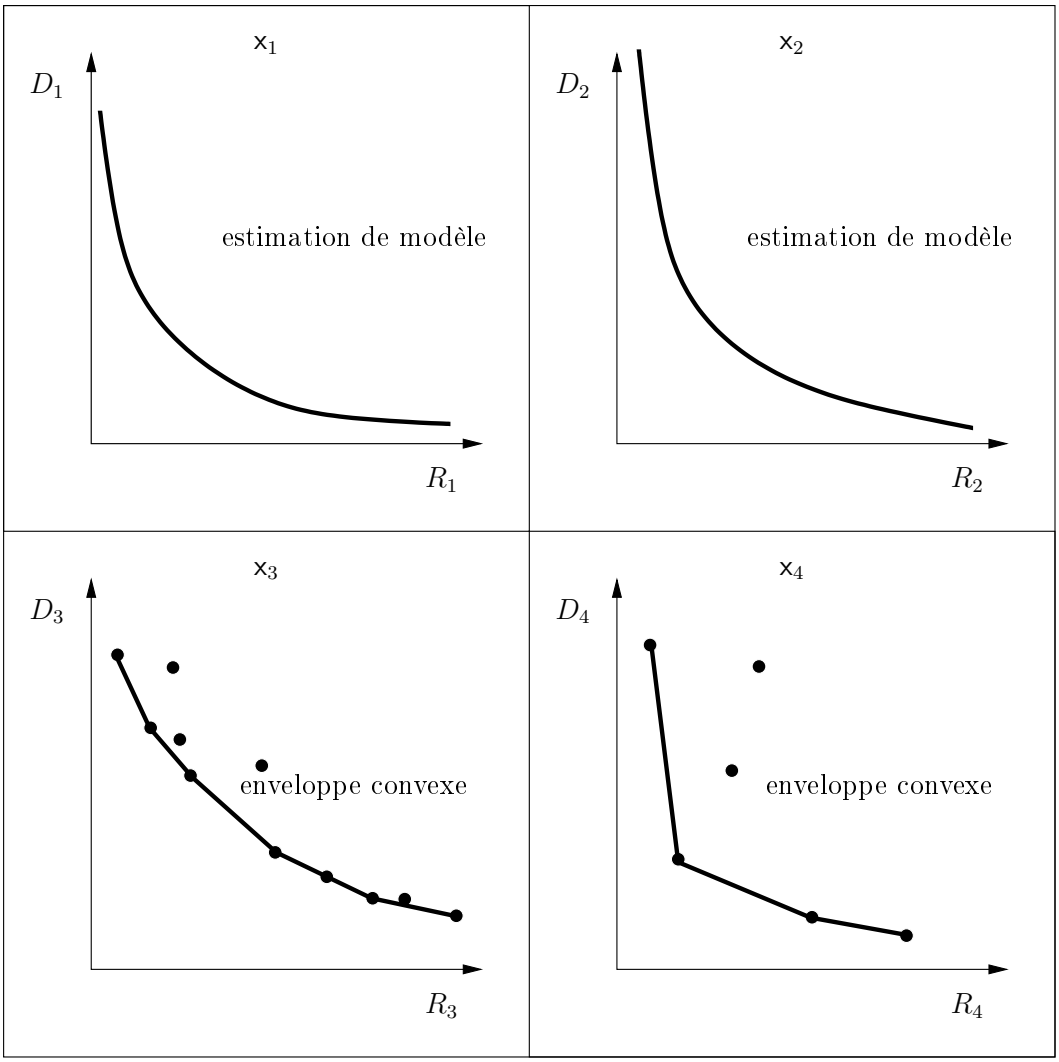

FIG. 1.16: Recherche des courbes débit-distorsion. Modèles statistiques [haut]. Enveloppe convexe [bas].

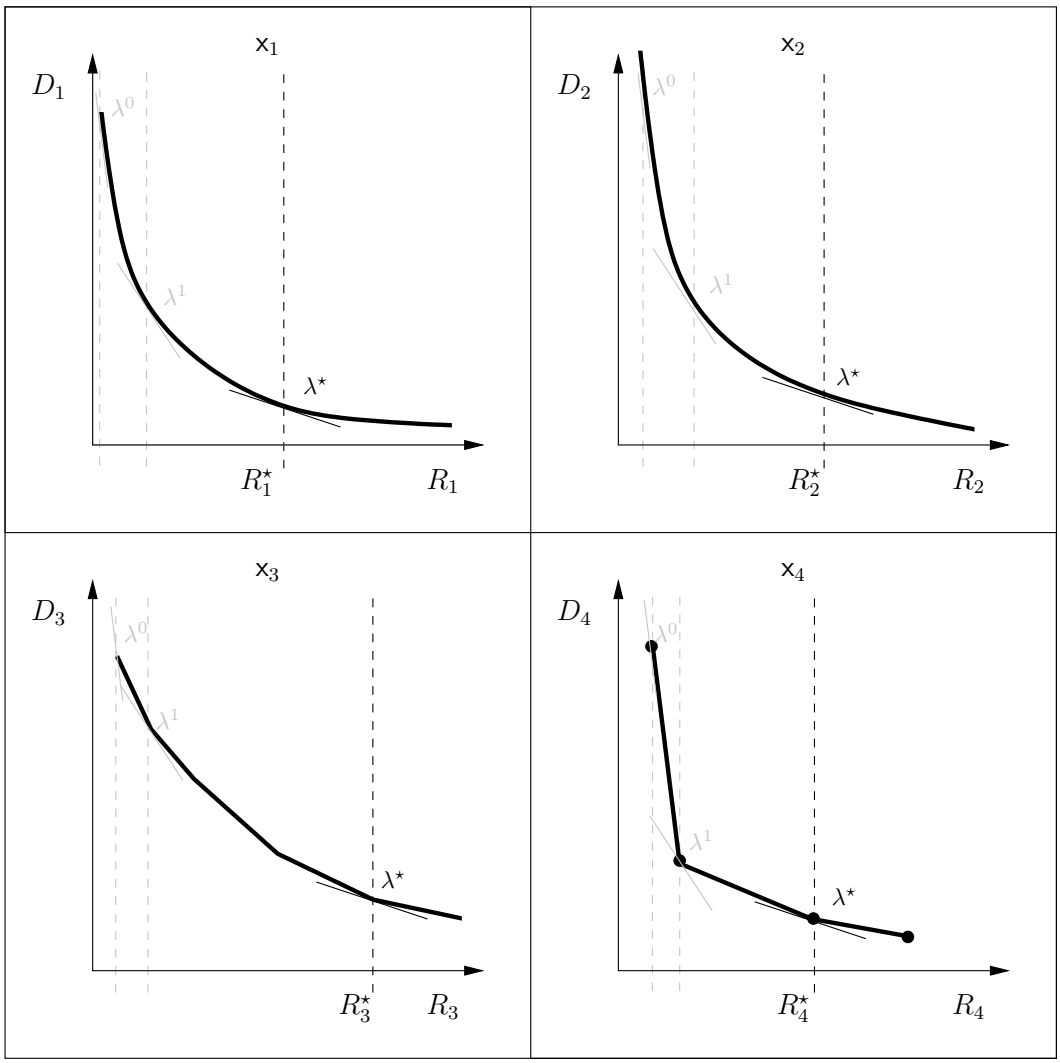

FIG. 1.17: Allocation de débit par recherche de la pente commune optimale $-\lambda^\star$ telle que la somme des débit $R_1 + R_2 + R_3 + R_4$ satisfasse la contrainte de débit R_t.

Chapitre 2

Etat de l'art

2.1 Transformées directionnelles

Depuis leur introduction il y a deux décennies, les ondelettes ont gagné un intérêt considérable en traitement du signal. L'idée de représenter un signal à différentes résolutions permet d'en extraire ses tendances principales en un nombre restreint de coefficients, tout en localisant précisément les discontinuités. Dans le contexte du traitement d'image, les ondelettes ont été utilisées pour des applications variées, telles que le débruitage et la compression, menant à des standards comme JPEG2000. Il est bien connu que les ondelettes sont optimales pour la représentation de signaux unidimensionnels (1D) possédant un nombre fini de discontinuités [30]. En effet, l'erreur quadratique moyenne d'une approximation non-linéaire faite à partir des k coefficients d'ondelette maximaux décroît en $O(k^{-1})$.

Dans le cas des images, pour des raisons de simplicité et d'efficacité, les ondelettes ont souvent été utilisées de manière séparable sur les axes horizontal et vertical. Il en résulte une décorrélation partielle de l'image, qui se traduit par la présence de nombreux coefficients de forte énergie le long des contours. Bien que la dépendance résiduelle soit néanmoins réduite et partiellement exploitée par les codeurs de sous-bandes dans le cas de la compression d'image, il peut sembler intéressant de chercher une transformée qui résolve ce problème en filtrant directement le long des contours de l'image.

Basées sur cette observation, de nombreuses transformées orientées, discrètes ou continues, ont été conçues récemment [31]. Certaines reposent sur des bancs de filtres directionnels fixes analysant l'image à des échelles, positions et orientations données. D'autres suivent une approche adaptative décrite par un modèle géométrique donnant explicitement la direction d'analyse locale. Nous allons dans ce chapitre faire un tour d'horizon des transformées directionnelles existantes, en les situant les unes par rapport aux autres et en présentant leurs principaux avantages. Le tableau 2.1 résume leurs différentes caractéristiques, tandis que la figure 2.4 représente le partitionnement fréquentiel réalisé par ces transformées.

D'autres approches, telles que les empreintes d'ondelettes [32], représentent efficacement les discontinuités en observant les dépendances entre les coefficients d'ondelettes

aux différentes échelles. Il est en effet montré dans [32] que pour des fonctions polynomiales de degré D, les coefficients d'ondelettes représentant une discontinuité forment un sous-espace vectoriel de dimension $D+1$ uniquement (appelé empreinte) dont il est possible d'extraire une base orthonormée par l'algorithme de Gram-Schmidt. Ainsi seuls $D+1$ coefficients sont nécessaires à la représentation de cette discontinuité, au lieu de l'ensemble des coefficients d'ondelettes concernés par cette discontinuité.

2.1.1 Approches non-adaptatives

Nous présentons tout d'abord les approches pour lesquelles la décomposition ne dépend pas du signal à analyser. Ces approches ont le mérite de ne pas nécessiter de surcoût pour spécifier, lors de la synthèse, la configuration utilisée à l'analyse. En revanche, la plupart de ces transformées ont l'inconvénient d'être redondantes, ce qui est désavantageux pour leur application en compression d'image.

2.1.1.1 Transformée de Radon

La transformée de Radon [33] consiste à projeter l'image sur un certain nombre d'orientations en intégrant l'image le long de la direction orthogonale à la projection (Fig. 2.1), puis à réaliser la transformée de Fourier de ces projections. La reconstruction s'obtient en plaçant, pour chaque orientation de projection choisie, les coefficients de Fourier obtenus le long de cette même orientation, dans le domaine fréquentiel. On obtient l'image reconstruite en effectuant ensuite une transformée de Fourier 2D inverse. La reconstruction parfaite pour cette transformée continue s'obtient pour un nombre de projections infini, parcourant l'ensemble des orientations possibles. La transformée de Hough [34, 35] est un cas particulier de transformée de Radon pour une image à valeurs binaire, et s'utilise principalement pour la reconnaissance de formes.

La transformée de Radon est très utilisée en tomographie, où les données capturées correspondent précisément à des projections du contenu de l'objet dont on cherche à obtenir une image. Une reconstruction approximative de l'image recherchée, d'autant plus précise que le nombre de directions de projection est élevé, est obtenue par transformée de Radon inverse. Notons que pour une image discrète carrée dont la taille est un nombre premier, la discrétisation proposée dans [36] peut s'appliquer pour obtenir une transformée de Radon discrète à reconstruction parfaite peu redondante. Une approximation discrète rapide (en $O(N^2 \log(N))$ opérations) et inversible de la transformée de Radon est présentée dans [37].

2.1.1.2 Ridgelets

Les *ridgelets* (lit. "crêtelettes") [38, 39] forment une extension naturelle de la transformée de Radon pour un nombre limité de directions, en se basant sur des fonctions d'ondelettes pour contrôler la précision en orientation et garantir la reconstruction parfaite. Étant donné une fonction d'ondelette $\psi(t)$, Candes propose de construire une base de fonctions :

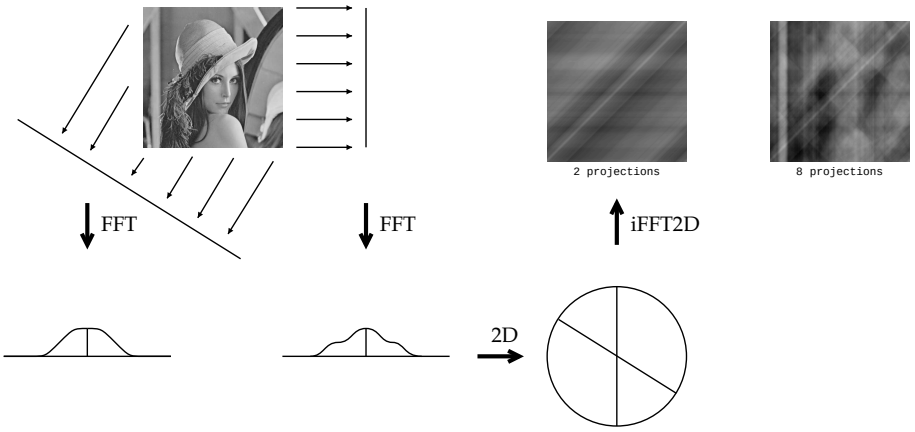

FIG. 2.1: Transformée de Radon.

$$R_{l,n,\theta}(\mathsf{t}) = M^{-\frac{l}{2}}\psi(\frac{(\cos(\theta),\sin(\theta))^\top \mathsf{t}}{M^l} - n),$$

où $l \in \mathbb{R}_+^\star$ représente l'échelle, $\theta \in [0, 2\pi]$ l'orientation et $n \in \mathbb{R}$ la position de la ridgelet. Cette famille est donc constituée de lignes d'orientation variable et dont la largeur dépend du support de l'ondelette choisie. Une dicretisation est également proposée dans [38] permettant d'obtenir une frame. Le facteur de redondance n'est pas donné de manière quantitative dans le cas général, et dépend du choix du facteur d'échelle M, et de la résolution angulaire et spatiale [38, p. 37].

Dans [40], une famille de ridgelets fenêtrées est proposée sous le nom d'ortho-ridgelets. Du fait du compromis sur l'incertitude temps-fréquence, ce fenêtrage implique une discrétisation en orientation pour une seule échelle donnée. Cependant, en partant d'une fenêtre de la taille de l'image et en considérant toutes les familles d'ortho-rigdelets définies sur le partitionnement dyadique de cette fenêtre initiale en sous-fenêtres, il est possible d'obtenir une décomposition en ridgelets multiéchelle. La famille totale obtenue est alors consistuée d'éléments dont le support est inscrit dans un rectangle de longueur l inférieure ou égale à la taille de l'image, et de largeur $0 < w < l$. Bien que pour chaque échelle, la famille d'ortho-ridgelets correspondante forme une frame, l'ensemble des fonctions ridgelets multiéchelle n'en est malheureusement pas une. D'une certaine manière, la famille de fonctions proposée par cette transformée est trop riche, dans le sens où la somme des énergies des coefficients n'est pas bornée pour tout signal d'énergie finie. Notons qu'il existe également une transformée en ridgelets finie, proposée dans [39], en se basant sur la discrétisation des transformées de Radon finies [36]. Il est remarquable de préciser que cette transformée est non-redondante, nécessitant toutefois d'avoir une image carrée dont la taille est un nombre premier.

2.1.1.3 Curvelets

La transformée en *curvelet* (lit. "courbelettes") a été proposée dans [41]. Cette transformée se dérive des ridgelets multiéchelles. Une première décomposition permet d'obtenir une analyse multiéchelles en sous-bandes centrées sur les couronnes de fréquences $\mathsf{f} \in [0, \frac{1}{2^{2s}}]^2 \backslash [0, \frac{1}{2^{2s+2}}]^2$, où $s \in \mathbb{N}$ représente l'échelle. Notons que cette découpe de l'espace fréquentiel n'est pas classique. Ces sous-bandes sont ensuites analysées par une transformée en orthoridgelets sur des blocs de taille $2^s \times 2^s$. Les atomes d'analyse sont alors des éléments rectilignes de taille $2^s \times 2^{2s}$ (Fig. 2.2). Ces éléments suivent donc une loi de changement d'échelle parabolique ($l = w^2$), bien adaptée à l'approximation des courbes [41, p. 15]. L'ondelette utilisée pour construire les orthoridgelets est l'ondelette de Meyer tandis que la fonction d'échelle de Lemarié est utilisée pour la représentation des basses fréquences.

Une construction différente et plus générale, reposant sur la théorie des frames, est présentée dans [42]. La frame d'analyse est construite directement à partir d'une fonction mère C bidimensionnelle de haute fréquence selon l'un des axes et de basse fréquence selon l'autre (typiquement le produit tensoriel d'une fonction d'ondelette et d'une fonction d'échelle). La famille de curvelets $(C_{l,\mathsf{n},\theta})_{l\in\mathbb{N},\mathsf{n}\in\mathbb{Z}^2,\theta\in\frac{2\pi}{2^l}\mathbb{Z}/2^l\mathbb{Z}}$ correspondante est alors donnée par :

$$C_{l,\mathsf{n},\theta}(\mathsf{t}) = 2^{\frac{3l}{2}} C(\mathsf{D}_l \mathsf{R}_\theta \mathsf{t} - \mathsf{n}),$$

où $\mathsf{D}_l = \begin{pmatrix} 2^{2l} & 0 \\ 0 & 2^l \end{pmatrix}$ est la matrice de sous-échantillonage du changement d'échelle (notons au passage que $2^{\frac{3l}{2}}$ correspond à la racine carrée de son déterminant), R_θ est la matrice de rotation d'angle θ et n indique la position de la curvelet. Cette transformée a été utilisée avec succès dans le cadre du débruitage [43].

2.1.1.4 Contourlets

Conçue directement dans le domaine discret, la transformée en contourlettes a été proposée dans [44, 45]. Cette transformée repose sur des filtres en éventails itérés pour l'analyse directionnelle [46] et une pyramide laplacienne pour l'analyse multirésolution [47]. Elle apporte une analyse similaire à la transformée en curvelets, tout en ayant l'avantage d'être très peu redondante et moins complexe. L'analyse multirésolution est effectuée à l'aide des filtres d'ondelette 9/7 [12]. Contrairement à la reconstruction classique de la pyramide laplacienne, l'opérateur de reconstruction utilisé est l'opérateur transposé de l'opérateur d'analyse, qui correspond approximativement à l'opérateur pseudo-inverse car ces filtres sont quasi-orthogonaux [4]. Les filtres en éventails servant à l'analyse directionnelle sont quant à eux issus de [7]. Un sous-échantillonnage sur lattice quinconce permet d'itérer ce banc de filtres pour obtenir les sous-bandes directionnelles sans redondance additionnelle. Ce filtrage est par ailleurs réalisable de manière séparable après décomposition polyphase. Nous reviendrons plus en détails sur cette transformée dans le chapitre 3, où nous proposerons différentes techniques permettant de l'utiliser pour la compression. Notons qu'un schéma appliquant la transformée directionnelle sur

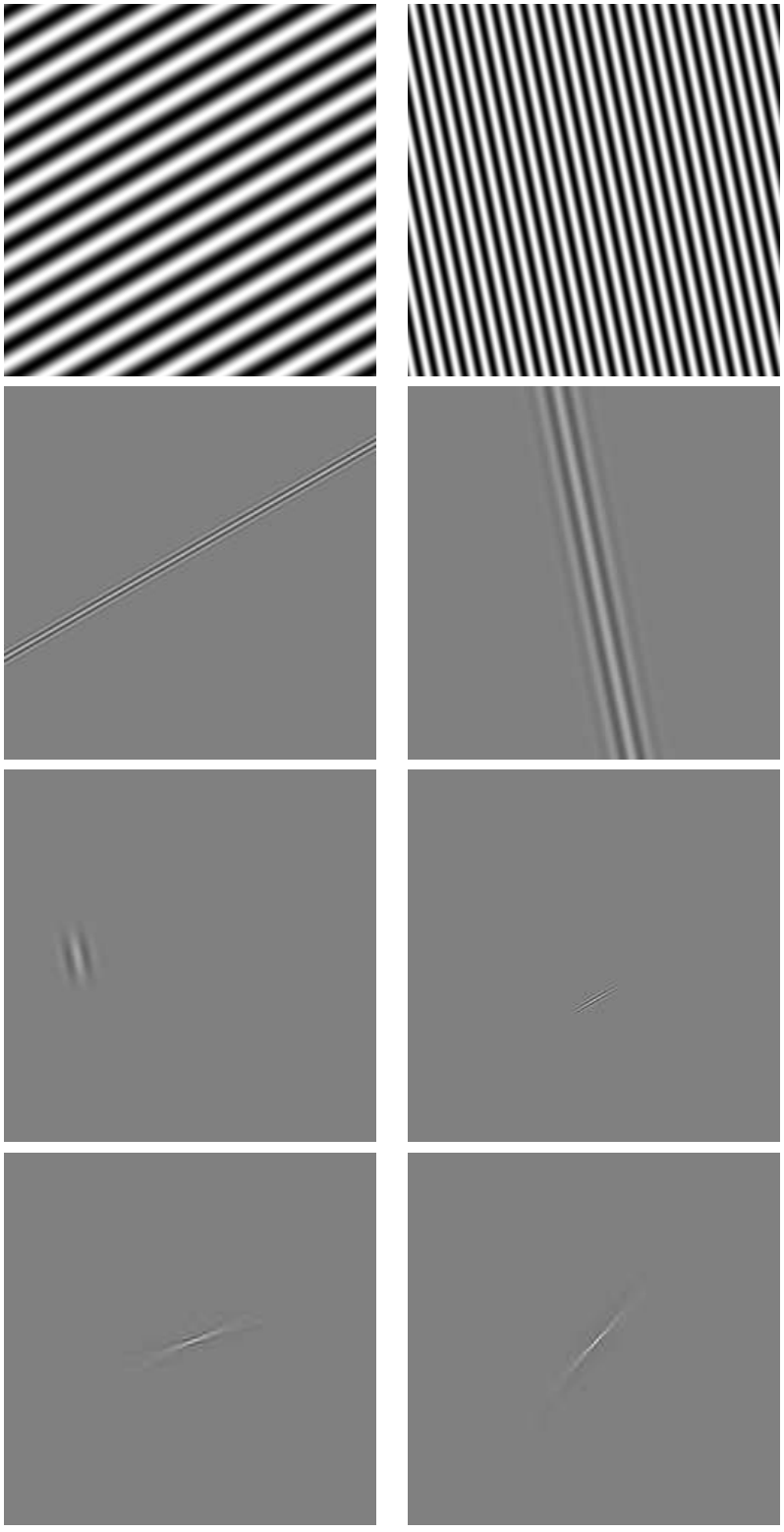

FIG. 2.2: Exemples de fonctions de base pour la transformée de Radon, les ridgelets, les orthoridgelets/curvelets et les contourlets (de haut en bas).

des sous-bandes d'ondelettes séparables a été présenté dans [48]. Ce codeur utilise un codeur de type SPIHT [49] modifié pour coder les sous-bandes directionnelles obtenues.

Dans [50], une version modifiée des contourlettes permettant de s'affranchir de la redondance est proposée. Cette transformée en contourlettes CRISP, repose sur une conception conjointe du banc de filtres directionnels et multirésolution. Les basses fréquences sont séparées en quatre sous-bandes tandis que les hautes fréquences sont séparées successivement en $3 \cdot 2^k$ sous-bandes ($k \in \mathbb{N}$). En plus de la paire de filtres en éventail, la conception de filtres en damier cisaillé, en parallélogramme, et en parallélogramme cisaillé [50, p. 9] est nécessaire à l'obtention de cette décomposition. En revanche, aucune application n'a pour l'instant été développée autour de cette transformée.

2.1.1.5 Ondelettes complexes

La transformée en ondelettes complexe, une autre transformée directionnelle redondante, est présentée dans [51, 52, 53, 54]. Les ondelettes complexes effectuent un filtrage différent dans les fréquences négatives et positives du signal. Il est alors possible avec une transformée en ondelettes complexes séparable de discriminer les signaux orientés selon une direction diagonale (+45°) des signaux orientés selon une direction antidiagonale (-45°). Contrairement aux ondelettes réelles, la transformée est itérée à la fois sur la bande basse correspondant à la partie réelle des coefficients et celle correspondant à la partie imaginaire des coefficients, ce qui mène à une double structure d'arbre, plutôt qu'une simple [53, p. 32]. Cette transformée fournit une analyse comparable aux pyramides orientables pour une redondance moindre et de meilleures propriétés de reconstruction [53, p. 35]. Enfin on peut voir la transformée en ondelettes complexes comme un cas particulier de transformée multi-ondelettes 2D [55] (en considérant un complexe comme un vecteur).

Cette transformée a été optimisée pour la compression dans [56] en utilisant une technique itérative basée sur la théorie de la projection sur ensembles convexes. Comme cette méthode est générale, elle peut également être utilisée pour réduire l'impact de la redondance sur les performances en compression d'autres transformées redondantes. Nous reviendrons sur ce point dans le chapitre 3 où nous appliquons cette technique à la transformée en contourlettes.

2.1.1.6 Transformée cortex

La transformée en cortex [57] provient du domaine de la vision par ordinateur. Elle consiste à séparer le plan fréquentiel en sous-bandes d'orientations et de fréquences particulières afin de modéliser le comportement du système visuel humain. En effet, il a été observé par les biologistes que les neurones du cortex visuel répondent à des orientations et des échelles particulières [58, 59]. Dans [57], Watson propose d'utiliser une décomposition en 4 orientations sur 4 niveaux. Daly [60] étend ensuite la transformée à 6 orientations sur 4 niveaux. Dans les deux cas, la transformée est réalisée en filtrant par produit dans le domaine fréquentiel, menant à une transformée hautement redondante.

Une structure de sous-échantillonage réduisant cette redondance est proposée dans [61] pour une application au codage d'image. Dans [62], une implémentation par pyramide laplacienne limitant la redondance à $\frac{4}{3}k$, où k est le nombre d'orientations, est proposée. Cette redondance reste toutefois rédhibitoire et cette transformée n'a pas donné lieu à des schémas de codage d'images compétitifs par rapport aux schémas basés sur des transformées en ondelettes séparables.

2.1.1.7 Pyramide orientable

La pyramide orientable [63, 64, 65, 66, 67] est une transformée redondante reposant sur la pyramide laplacienne [47]. A l'exception de la bande de basse fréquence, les sous-bandes sont filtrées selon un nombre d'orientations choisi en fonction de l'application. Ces filtrages correspondent aux calculs des dérivées jusqu'à un certain ordre. Pour un nombre d'orientation k, il est alors possible d'obtenir les dérivées d'ordre $k-1$ de l'image en tout point en tant que combinaison linéaire des dérivées selon chaque direction (par interpolation). Une construction itérative des filtres directionnels dans le domaine fréquentiel est proposée dans [64]. Dans [68] la redondance inhérente à la pyramide orientable est exploitée pour réduire l'erreur de reconstruction due à la quantification.

2.1.2 Approches adaptatives

Nous présentons à présent les approches dépendantes du signal, qui offrent en général une plus grande flexibilité que les approches non adaptatives. Le signal est analysé moyennant une information adjacente permettant de sélectionner les représentants adéquats dans un dictionnaire de fonctions très redondant. Cette sélection est souvent réalisée de sorte à obtenir une base, afin de limiter la redondance à l'information adjacente, de nature très différente des coefficients de transformée.

2.1.2.1 Matching pursuit

L'algorithme de Matching pursuit [69] consiste à rechercher successivement un sous-ensemble de vecteurs issus d'un dictionnaire structuré et dont la combinaison linéaire représente au mieux le signal. À chaque étape, le signal est projeté sur l'ensemble des vecteurs du dictionnaire. La projection de plus forte amplitude ainsi que le vecteur correspondant sont conservés pour représenter le signal. Cette composante est ensuite soustraite au signal, formant un nouveau signal d'erreur résiduelle pour l'étape suivante. Il s'agit donc d'une optimisation locale, bien adaptée à la représentation de signaux non-stationnaires. D'un point de vue codage, cette technique est utile pour sélectionner les vecteurs d'intérêt dans une famille fortement redondante d'atomes temps-fréquence. Si l'on considère également la quantification des projections sur les vecteurs sélectionnés, cette technique est très similaire à une quantification vectorielle par gain et forme [70, 71].

2.1.2.2 Paquets d'ondelettes

Dans les techniques par paquets d'ondelettes [72, 73], l'ensemble des décompositions par ondelettes dyadiques du signal est considéré (au lieu d'itérer la décomposition uniquement sur la bande basse). Dans les application de codage, la base offrant le meilleur compromis débit-distorsion est sélectionnée [74] par optimisation lagrangienne. Le coût de codage de la base choisie est pris en compte lors de cette minimisation.

2.1.2.3 Brushlets

La transformée en brushlets a été proposée par Meyer et Coifman dans [75]. Elle donne un partitionnement adaptatif du plan fréquentiel offrant plus de flexibilité que les paquets d'ondelettes en s'affranchissant de la contrainte de séparabilité. Il est alors possible de représenter un motif orienté à l'aide d'un seul coefficient. La décomposition est effectuée en appliquant successivement des opérateurs de fenêtrage à l'image suivis chacun d'une transformée de Fourier. La reconstruction s'obtient alors en effectuant la transformée inverse de chaque composante puis en appliquant l'opérateur de fenêtrage dual et en sommant finalement chaque contribution. Des constructions de bases orthogonales et bi-orthogonales sont proposées dans le cas 1D et 2D, ainsi qu'une procédure de discrétisation. Une optimisation par quad-tree est mise en oeuvre pour sélectionner le partitionnement fréquentiel adéquat dans le cadre de la compression d'image. Une comparaison honnête, incluant le processus de codage des sous-bandes, est présentée par rapport à la transformée en ondelettes séparables couplée au codeur EZW [76]. Ce schéma montre des performances toutefois assez décevantes d'environ 0 à -3 dB.

2.1.2.4 Beamlets

La décomposition en beamlets [77] considère un partitionnement de l'image en quad-tree, puis effectue une transformée de Radon dans chaque bloc. Les coefficients de beamlets sont liés par une relation multiéchelle, où chaque beamlet à un niveau donné est décomposée en trois beamlets connexes au niveau suivant. Cette transformée permet d'approximer les courbes dans les images et d'en extraire les contours par sélection dans le graphe de connexité des beamlets.

2.1.2.5 Wedgelets

La décomposition en wedgelets [78] représente une image par un quad-tree dans lequel chaque bloc est séparé en deux régions d'intensité différentes par une ligne. Elle est donc, en quelque sorte, duale à la décomposition en beamlets en considérant les intégrales de l'image de chaque coté du segment représentant le contour plutôt que l'intégrale le long du segment. Bien que cette décomposition fournisse une approximation assez grossière des images, elle a été combinée dans [79] avec une décomposition en ondelettes séparables et un codeur SFQ [80] pour la compression. Cette transformée a été généralisée en dimension plus élevée sous le nom de surflets [81]. Enfin, les platelets

[82] sont une extension des wedgelets considérant un modèle affine de l'intensité des régions des deux cotés du segment de contour.

2.1.2.6 Bandelettes

La transformée en bandelettes repose sur un modèle géométrique explicite de l'image pour effectuer une analyse orientée le long des contours. Dans une première approche [83], les contours sont représentés par des courbes paramétriques le long desquelles une ondelette séparable est déformée. La décomposition en ondelettes est itérée également le long du contour pour exploiter la forte corrélation existant dans cette direction. Un calcul de gradient multirésolution permet d'extraire les contours et de sélectionner un certain nombre de courbes paramétriques codées par ondelettes également. Les coefficients de bandelettes le long de ces courbes sont codés ensuite et leur contribution est ôtée de l'image. L'erreur résiduelle est finalement codée par ondelettes séparables. Il est néanmoins délicat dans cette approche d'effectuer l'allocation de débit entre la géométrie, les coefficients de bandelettes et les coefficients d'ondelettes de manière efficace.

Une seconde approche [84] considère un champ d'orientations plutôt qu'un nombre restreint de courbes. Ce champ, représenté sous forme de quad-tree, indique l'orientation locale à l'intérieur de blocs de l'image. Lorsqu'aucune orientation ne semble pertinente, une décomposition en ondelettes séparables est également possible à l'intérieur d'un bloc. Sinon, une décomposition en bandelettes le long de ces orientations est effectuée. Ce champ peut alors être optimisé en terme de débit-distorsion pour réaliser l'allocation de débit entre les informations de géométrie et de texture. Le calcul des coefficients de bandelettes s'effectue de manière séparable en filtrant le long du champ d'orientation, à l'aide de techniques de lifting sur grilles irrégulières [85], puis dans une direction fixe où le champ est supposé constant (Fig. 2.3).

La complexité principale de cette transformée réside dans la recherche du champ d'orientation et l'optimisation débit-distorsion pour répartir le débit entre les différentes informations. Cette technique a également été adaptée à la compression de surfaces [86].

2.1.2.7 Directionlets

Une transformée discrète à échantillonage critique appellée transformée en *directionlets* est présentée dans [87, 88]. Un filtrage séparable est effectué le long de deux directions d'analyse correspondant aux vecteurs de base d'une lattice, dont le déterminant donne le nombre de sous-bandes. Cette technique fournit une analyse anisotrope de l'image, bien adaptée aux images à forte structure géométrique. En effectuant un partitionnement adaptatif et en appliquant une transformée en directionlets appropriée, fonction de la courbure locale des contours, il est possible d'obtenir une meilleure approximation qu'avec des ondelettes séparables. Notons que ces dernières peuvent être vues comme un cas particulier de directionlets sur une lattice carrée.

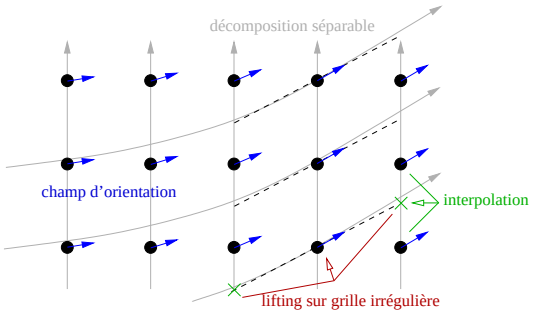

FIG. 2.3: Décomposition en bandelettes obtenue par déformation d'une ondelette séparable le long du champ d'orientations. La décomposition s'effectue de manière séparable en appliquant successivement les étapes lifting de l'ondelette le long de la direction locale, puis en appliquant une décomposition verticale. Une décomposition sur plusieurs niveaux est effectuée le long du contour avant de passer à la décomposition verticale. Le champ d'orientations est supposé constant le long de la direction verticale.

2.1.3 Comparaison des différentes transformées

Parmi l'ensemble de ces transformées directionnelles, seul un nombre très restreint a été appliqué à la compression. En effet, la plupart de ces transformées ont le désavantage d'être redondantes. Ceci implique qu'un nombre plus important de coefficients doit être codé par rapport aux transformées à échantillonage critique à haut débit, ce qui se traduit généralement en un coût supplémentaire en débit. À bas débit, ce surcoût peut être compensé par la meilleure capacité des transformées orientées à représenter les contours de l'image, permettant d'obtenir une distorsion plus faible que les transformées non orientées pour le même nombre de coefficients. Pour le débruitage en revanche, la redondance n'est pas problématique, ce qui explique le succès de ces transformées dans ce cadre.

Le tableau récapitulatif 2.1 permet de comparer les transformées orientées présentées ici en termes de redondance et de nombre d'orientations offertes. À chaque transformée orientée correspond un partitionnement du plan fréquentiel différent, que la figure 2.4 illustre sur un certain nombre d'exemples. Les approches discrètes fournissant la meilleure sélectivité directionnelle sont la pyramide orientable et la transformée en contourlettes.

Les premières transformées directionnelles à avoir vu le jour sont les transformées de Radon et la transformée en cortex. Malheureusement, ces transformées étant définies dans le domaine continu, leur discrétisation mène à une redondance très élevée. Elles ont toutefois été source d'inspiration pour des transformées plus récentes, comme la famille des transformées en ridgelets et curvelets, qui revient à utiliser une fonction d'ondelette pour discrétiser en orientations la transformée de Radon. La famille des transformées en contourlettes et la pyramide orientable s'inspirent en revanche du partitionnement fréquentiel proposé par la transformée en cortex. La transformée en contourlettes présente de plus l'intérêt notable d'être peu redondante, ce qui en fait une bonne candidate pour son application à la compression. Nous verrons cependant au

chapitre 3 que même cette faible redondance peut s'avérer problématique à haut débit. Enfin, les approches par ondelettes complexes et par paquets d'ondelettes étendent le principe de la transformée en ondelettes séparables pour proposer d'autres partitionnements fréquentiels plus adaptés au signal.

Les approches adaptatives présentent quant à elles l'avantage de fournir des transformées non redondantes, moyennant une information adjacente permettant de spécifier la configuration de codage choisie. Cette séparation de l'information entre un modèle d'image et des coefficients associées est très nette dans la technique de matching pursuit où l'information adjacente est donnée par l'index de l'atome choisi, et le coefficient associé par la projection du signal sur ce même atome. Le surcoût induit par le codage de l'information adjacente doit cependant être pris en compte pour répartir le débit entre cette information supplémentaire et l'information représentant les coefficients de la transformée. Dans la transformée en bandelettes par exemple, l'information adjacente consiste en un champ dense d'orientations, qui est optimisé selon des critères débit-distorsion pour en minimiser le coût de codage. Enfin, certaines techniques adaptatives, comme la famille des wedgelets, qui représentent l'image de manière assez simpliste, peuvent être combinée efficacement à d'autres transformées non orientées, pour effectuer un codage en deux passes. Une première passe extrait les informations de contour de l'image par projection sur la famille de fonctions associée à la transformée directionnelle, puis une seconde passe représente l'erreur résiduelle au moyen d'une transformée classique.

La plupart des transformées discrètes présentées ici considèrent soit des approches dans le domaine de Fourier, comme la transformée en brushlets et la transformée en cortex, soit des filtrages séparables dans le domaine spatial, comme la transformée par paquets d'ondelettes. Toutefois, deux exception notables que sont les contourlettes et les directionlettes, proposent une structure d'échantillonage non séparable. En se plaçant dans le cadre de la théorie des lattices et du filtrage multidimensionnel, ces approches offrent à la fois une analyse discrète directionnelle et peu redondante. Nous pensons qu'il s'agit là d'une voie prometteuse pour la conception de transformées directionnelles adaptées à la compression, comme la transformée en ondelettes orientée que nous proposons au chapitre 4.

2.2 Codage des sous-bandes

Le but des transformées présentées dans la section précédente est de décorreler les données brutes de l'image représentées par ses pixels. Cette décorrélation n'est cependant pas parfaite et les coefficients obtenus après transformée restent dépendants statistiquement. Ainsi, bien qu'une loi gaussienne généralisée puisse représenter avec fidélité la statistique de premier ordre des sous-bandes, seuls les codeurs exploitant l'information mutuelle résiduelle entre les coefficients ont permis d'obtenir des performances bien meilleures que les codeurs précédents basés sur de la quantification vectorielle. De plus, les transformées en ondelettes offrant naturellement une représentation progressive de l'image, il est intéressant de conserver cette propriété lors du codage des sous-bandes. Ainsi, dans les codeurs emboîtés (embedded), la quantification et le codage sont égale-

Transformée	multiéchelle	adaptative	orientations	redondance
Ondelettes séparables	oui	non	3	1
Paquets d'ondelettes [72, 73]	oui	oui	3	1
Transformée de Radon [33]	non	non	∞	-
Transformée de Radon finie$^{\{3\}}$ [36]	non	non	$p+1$	$1+\frac{1}{p}$
Ridgelets [38]	non	non	∞	-
Ortho-ridgelets$^{\{4\}}$ [40]	non	non	k	-
Ridgelets finies$^{\{3\}}$ [39]	non	non	$p+1$	1
Curvelets$^{\{4\}}$ [41, 42]	oui	non	2^k	-
Contourlettes$^{\{4\}}$ [4, 44, 45]	oui	non	2^k	$\frac{4}{3}$
Contourlettes CRISP$^{\{4\}}$ [50]	oui	non	$3 \cdot 2^k$	1
Ondelettes complexes [51, 52, 53, 56]	oui	non	6	4
Directionlettes [87]	oui	oui	1	1
Transformée cortex [57, 60]	oui	non	4 ; 6	$\frac{16}{3}$; 8
Pyramide orientable$^{\{4\}}$ [63, 64, 65, 66, 67]	oui	non	k	$\frac{4}{3}k$
Brushlets [75]	oui	oui	$^{\{2\}}$	1
Beamlets [77]	oui	oui	∞	-
Wedgelets [78, 79]	oui	oui	∞	-
Matching pursuit [69]	oui	oui	$^{\{1\}}$	$^{\{1\}}$

$^{\{1\}}$: au choix en fonction de l'ensemble d'atomes choisis.
$^{\{2\}}$: au choix en fonction du partitionnement fréquentiel par quad-tree.
$^{\{3\}}$: sur $(\mathbb{Z}/p\mathbb{Z})^2$, avec p premier.
$^{\{4\}}$: avec $k \in \mathbb{N}$.
- : transformée continue dont la redondance est variable en fonction de la discrétisation choisie.

TAB. 2.1: Tableau récapitulatif des transformées orientées.

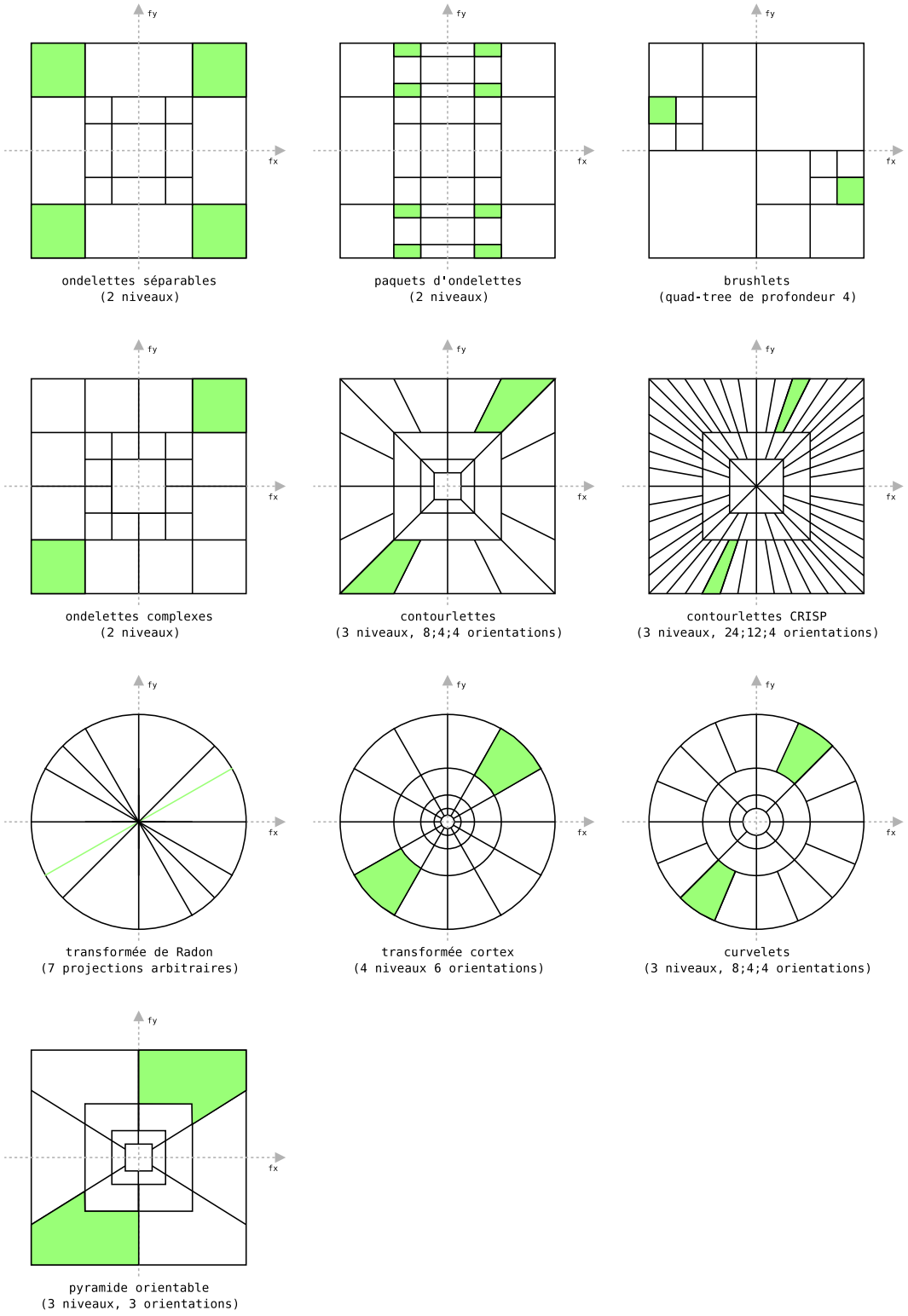

FIG. 2.4: Partitionnement du plan fréquentiel pour diverses transformées directionnelles en supposant les filtres idéaux. Pour chacune de ces décompositions, l'une des sous-bandes est représentée en vert clair.

ment réalisés de manière progressive, en commençant par coder partiellement les coefficients de plus forte amplitude, puis en raffinant la quantification de ces derniers et en en codant de nouveaux. Nous commençons par présenter les codeurs non progressifs, puis les codeurs emboîtés, basés sur des structures d'arbres ou de blocs.

2.2.1 EQ

L'algorithme d'Estimation-Quantization [89] (EQ) consiste à modéliser les coefficients de chaque sous-bande par un mélange de gaussiennes généralisées, dont la variance est conditionnée par un voisinage causal et le paramètre de forme est fixe. L'expression de la gaussienne généralisée est donnée par la formule suivante :

$$\mathbb{P}(X = x) = \frac{\alpha\beta}{2\Gamma(\frac{1}{\beta})} e^{-|\alpha x|^{\beta}},$$

où β est le paramètre de forme, et α est donné en fonction de la variance σ^2 par $\alpha^2 = \frac{\Gamma(\frac{3}{\beta})}{\sigma^2\Gamma(\frac{1}{\beta})}$. Cette loi se réduit à l'expression d'une laplacienne lorsque $\beta = 1$ et d'une gaussienne lorsque $\beta = 2$. Elle représente de manière assez fidèle la distribution stationnaire des coefficients pour un paramètre β autour de $0.6 - 0.7$ comme l'illustre la figure 2.5. Dans l'algorithme EQ, ce paramètre est estimé pour chaque sous-bande à l'encodeur et transmis au décodeur. La variance est quant à elle estimée localement par maximum de vraisemblance, en fonction de la valeur de coefficients précédemment quantifiés dans un voisinage causal du coefficient considéré. Lorsque tous les coefficients voisins sont quantifiés à zéro, la variance du coefficient est imprévisible et une valeur par défaut est utilisée. Cette variance par défaut est également estimée à l'encodage et transmise au décodeur.

Une fois les paramètres de la gaussienne généralisée estimés pour le coefficient courant, celui-ci est quantifié pour le débit cible donné. Un quantificateur uniforme par zone morte est sélectionné parmi un ensemble de quantificateurs prédéfinis indexés par la pente $-\lambda$ correspondante sur la courbe débit-distorsion. Les probabilités des symboles quantifiés sont également stockées dans cette table pour l'étape de codage entropique.

Ce codeur a l'avantage d'être simple et de complexité très réduite, offrant des performances très satisfaisantes. Il a par contre l'inconvénient de ne pas permettre un décodage progressif efficace de l'image, car chaque coefficient est codé totalement avant de passer au suivant.

2.2.2 SFQ

La quantification espace-fréquence [80] (Space-frequency quantization, SFQ) sépare les coefficients d'ondelettes en deux classes. En effet, les transformées citées précédemment ont la caractéristique de concentrer l'énergie du signal en majeure partie dans les basses fréquences mais également dans des paquets de coefficients haute-fréquence localisés dans l'espace (phénomène de clustering). Ainsi, leur modélisation par une loi

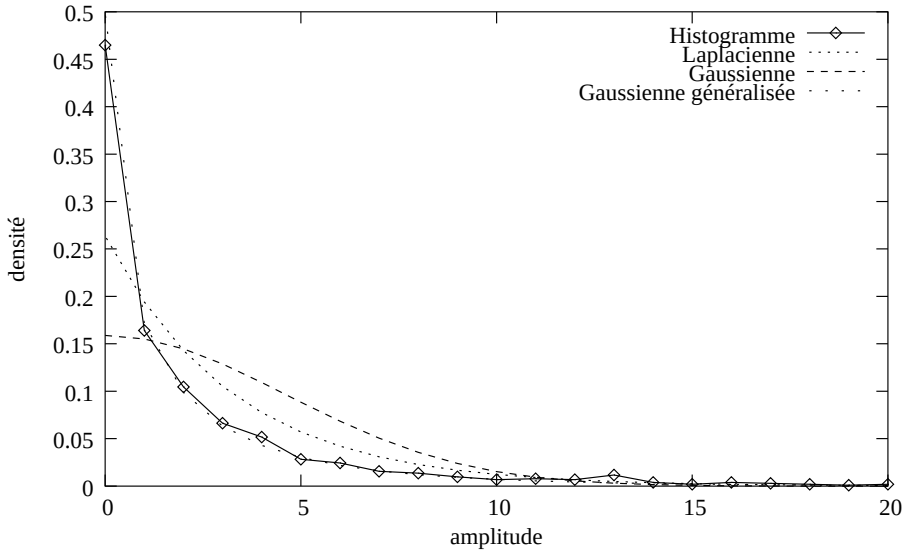

FIG. 2.5: Modèles stationnaires de coefficients de sous-bandes d'ondelettes. L'histogramme représente l'amplitude des coefficients quantifiés de la sous-bande diagonale du 3ème niveau d'une décomposition en ondelette séparable de l'image *lena*. Les modèles gaussien, laplacien et gaussien généralisé avec $\beta = 0.6$ sont respectivement à une distance de Kullback-Leibler de 0.58, 0.18 et 0.03 bits/symbole de cet histogramme.

stationnaire unique par sous-bande, correspondant à l'hypothèse d'indépendance spatiale, est loin d'être représentative de la statistique réelle des coefficients.

Le succès des techniques de codage de sous-bandes basées sur des structures d'arbre provient du fait qu'elles tiennent compte de cette dépendance statistique en représentant des groupes de coefficients nuls à l'aide d'un seul symbole appelé *zerotree*. Il s'agit donc d'une sorte de quantification vectorielle des coefficients insignifiants permettant de prendre en compte leur dépendance. Les autres coefficients sont traités indépendamment et quantifiés par un quantificateur scalaire uniforme suivi d'un codeur entropique.

L'allocation de débit entre les zerotrees, qui par complémentarité codent la position des paquets de coefficients énergétiques, et les coefficients non-nuls est réalisée de manière conjointe. Une optimisation débit-distorsion est réalisée afin de déterminer les configurations optimales de zerotree pour un point de débit cible donné. Ainsi, pour une pente $-\lambda$, et un pas de quantification Δ fixés, la configuration de chaque arbre ayant pour racine un coefficient de la bande basse est déterminée par une approche ascendante. Les coefficients sont agrégés en zerotree si le compromis débit-distorsion (représenté par un lagrangien) de cette représentation, à savoir un coût supposé nul en débit et une distorsion égale à l'énergie des coefficients, est plus intéressant que la quantification individuelle des coefficients de l'arbre. Comme le coût de codage des zerotree n'est en réalité pas nul, une correction a posteriori est effectuée. En itérant sur l'ensemble des pas de quantification Δ possibles, et en effectuant une recherche dichotomique de la pente λ satisfaisant la contrainte de débit, la configuration globale optimale est

déterminée. Un codeur arithmétique adaptatif est utilisé pour réaliser le codage final des symboles.

Un résultat intéressant de cette technique est de montrer que la distribution des coefficients significatifs est globalement identique dans chaque sous bande, une fois les coefficients de faible amplitude codés par zerotree. Ils ont de plus une distribution relativement plate, justifiant l'utilisation d'un quantificateur uniforme. Toutefois, ainsi qu'il est suggéré dans [80], le gain de codage dû à une meilleure répartition des centroïdes en dimension finie pourrait être exploité par une quantification vectorielle des coefficients significatifs (comme par exemple une quantification TCQ contrainte en entropie).

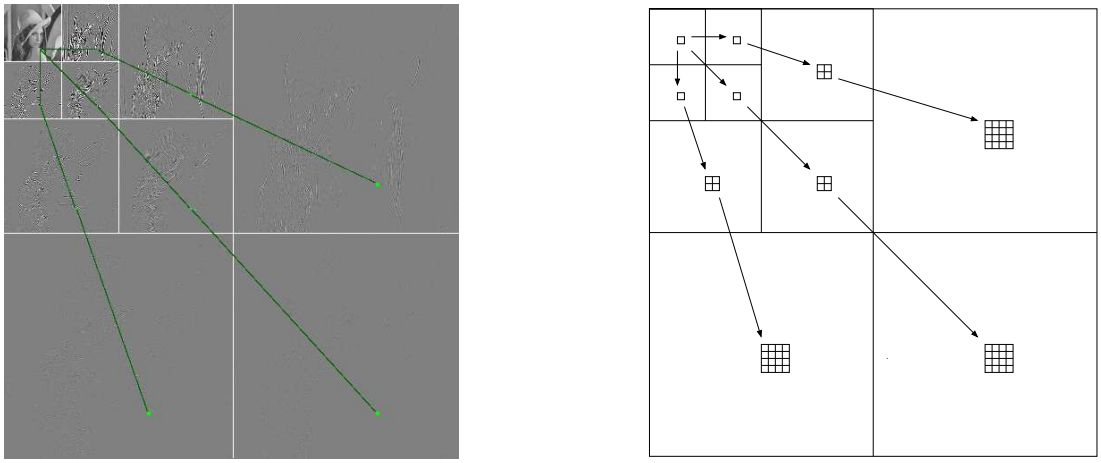

FIG. 2.6: Structure d'arbre des coefficients d'ondelette.

2.2.3 EZW

Le premier codeur en sous-bande par zerotree à avoir été introduit est le codeur EZW [76]. Ce codeur offre une représentation progressive de l'image tout en apportant d'excellente performances débit-distorsion par rapport à des codeurs bien plus complexes et non progressifs de l'époque [12] [90] [91].

Les coefficients sont parcourus dans l'ordre classique colonne puis ligne, des sous-bandes basses aux sous-bandes hautes. Le codage s'effectue progressivement en deux passes successives. Étant donné un seuil initial transmis au décodeur, la première passe de *signifiance* consiste à coder chaque coefficient à l'aide d'un symbole parmi quatre, notés 'P', 'N', 'T' et 'Z'. Si l'amplitude du coefficient est supérieure au seuil, un symbole 'P' ou 'N' est transmis en fonction du signe respectivement positif ou négatif du coefficient. Dans ce cas, ce coefficient est inséré dans la liste des coefficients significatifs (LSP) une fois son amplitude réduite de la valeur du seuil, et supprimé de la représentation par ondelettes en y étant annulé. Dans le cas contraire, l'ensemble des coefficients appartenant à l'arbre (Fig. 2.6) dont la racine est le coefficient courant est considéré. Si tous ces coefficients sont d'amplitude inférieure au seuil, un symbole 'T' est transmis,

indiquant la présence d'un zerotree. Sinon, un symbole 'Z' est transmis, correspondant à un coefficient insignifiant isolé à la position courante. Lors du parcours, les coefficients identifiés comme appartenant à un zerotree précédemment codé sont omis. Une fois la passe de signifiance effectuée, le seuil est divisé par deux.

La seconde passe, dite de *raffinement*, permet de préciser la valeur des coefficients significatifs (stockés dans la LSP) en comparant leur amplitude courante au seuil. Si celle-ci est supérieure au seuil, un symbole '1' est transmis et l'amplitude du coefficient est diminuée de la valeur du seuil, sinon un symbole '0' est transmis. Cette liste est ensuite triée par ordre lexicographique décroissant des symboles transmis pour chaque coefficient ('1' > '0', '11' > '1') pour améliorer l'adaptation du codeur entropique (bien que ce gain ne soit pas très significatif en pratique). Ces passes sont itérées jusqu'à obtenir un seuil inférieur au coefficient d'amplitude minimale. Le seuil initial étant choisi pour être une puissance de deux, les opérations de comparaison, de soustraction et de divisions sont réalisées très rapidement à l'aide d'opérations logiques. En effet, à chaque passe, l'algorithme ne considère que l'un des bits de chaque coefficient, en commençant par le bit de poids fort. C'est pour cette raison que cette technique se nomme également codage par *plans de bits*.

Le codage entropique des symboles produits est réalisé à l'aide d'un codeur arithmétique adaptatif. Pour le codage de la signifiance, le codeur arithmétique utilise un alphabet à quatre symboles, et considère quatre *contextes* donnés par la signifiance du coefficient parent et du coefficient à gauche du coefficient considéré. Ces contextes permettent d'estimer de manière adaptative quatre lois conditionnelles différentes. Le gain de codage dû à l'utilisation de plusieurs contextes est de l'ordre du dixième de dB, au dépend d'une complexité accrue. Notons que les codeurs actuels, comme nous le verrons par la suite, utilisent généralement un nombre de contextes bien plus important. Pour le codage du raffinement, l'alphabet est constitué de deux symboles et un contexte unique est utilisé.

Du fait de l'utilisation d'une technique de quantification emboîtée, le flux produit est décodable de manière progressive. Il est ainsi possible de le tronquer arbitrairement pour adapter le débit sans empêcher le décodage, l'image obtenue étant bien entendu de qualité plus faible. Il est également possible d'arrêter le décodage en cours si des contraintes de délai ou de coût de calcul sont à respecter. Ce codeur fournit donc une représentation compressée de l'image progressive à la fois en qualité et en complexité.

2.2.4 SPIHT

L'algorithme SPIHT [49] reprend les principes évoqués dans EZW tout en proposant de partitionner récursivement les arbres de coefficients. Ainsi, là où EZW codait un coefficient non significatif isolé ('Z'), SPIHT effectue un partitionnement récursif de l'arbre de manière à déterminer la position des coefficients significatifs dans la descendance du coefficient considéré. Les coefficients significatifs sont codés de manière similaire à EZW : leur signe est envoyé dès qu'ils sont identifiés comme étant signifiants et ils sont ajoutés à la liste des coefficients à raffiner. Cet algorithme fonctionne également par plans de bits. Il offre des performances remarquables, atteignant celles d'EZW *sans*

codage entropique. En ajoutant un codage entropique de l'information de signifiance, un gain supplémentaire entre 0.3 et 0.6 dB est obtenu.

Les bits envoyés lors de la passe de signifiance correspondent au programme exécuté à l'encodeur lors de l'exécution de l'algorithme de classement en coefficients significatifs et non significatifs. En suivant le même programme, le décodeur reste synchrone avec les décisions de l'encodeur et retrouve la même classification. Cet algorithme repose sur la gestion de trois listes, de coefficients signifiants (LSP), de coefficients insignifiants (LIP) et d'ensembles insignifiants (LIS). Moyennant un seuil de signifiance divisé par deux à chaque itération, et dont la valeur initiale est transmise au décodeur, l'algorithme se déroule de la manière suivante. La liste des coefficients signifiants est initialement vide, tandis que la liste de coefficients insignifiants contient les racines de chaque arbre (coefficients de la bande basse) et la liste d'ensembles insignifiants contient l'ensemble des descendants de chaque arbre. Cette partition initiale est segmentée récursivement au moyen de deux règles. Si un ensemble de descendants d'un noeud est signifiant, il est séparé en quatre coefficients fils directs de ce noeud, et l'ensemble des autres descendants. Les fils directs sont ajoutés à la LIP ou la LSP en fonction de leur signifiance. Si au moins un élément de l'ensemble des autres descendants est signifiant, cet ensemble est séparé en quatre ensembles insignifiants ajoutés à la LIS. Le fait de traiter les coefficients par groupes de quatre permet d'effectuer un codage entropique efficace par la suite. Comme dans EZW, la passe de raffinement consiste à coder progressivement les bits de poids faibles des coefficients significatifs.

Lorsque l'accroissement de complexité dû à l'utilisation d'un codeur entropique n'est pas limitant, un codage arithmétique adaptatif contextuel des bits de signifiance est envisageable pour améliorer les performances en compression. Les coefficients étant codés par groupes de quatre, il est intéressant de les traiter globalement pour exploiter une entropie d'ordre supérieur à 1. Les coefficients pouvant uniquement passer de l'état insignifiant à l'état signifiant, la taille de l'alphabet nécessaire pour représenter ces changements varie en fonction du nombre de coefficients déjà signifiants dans le groupe. Ainsi, il est proposé dans [49] d'utiliser quatre contextes en fonction du nombre de coefficients insignifiants variant de 1 à 4, conditionnant respectivement une loi sur un alphabet de 2 à 16 symboles. Les bits de signe ainsi que les bits de raffinement ne sont pas compressés, leur entropie étant déjà proche de 1.

2.2.5 SPECK

Offrant des performances comparable à SPIHT, l'algorithme SPECK [92] exploite des structures d'ensembles de coefficients non signifiants en blocs plutôt qu'en arbres. Ces structures de blocs permettent de s'affranchir efficacement de la non stationnarité des coefficients en adaptant localement la statistique utilisée pour le codage.

Les coefficients sont initialement séparés en deux ensembles, l'un noté $\mathcal{S}$ contenant les coefficients de basses fréquences et l'autre, noté $\mathcal{I}$ contenant les autres coefficients. De la même manière que dans SPIHT, deux listes sont tenues à jour, pour représenter les coefficients significatifs (LSP) et les ensembles de coefficients non significatifs (LIS). La liste d'ensembles non significatifs contient des blocs de coefficients de taille variable,

y compris les coefficients isolés vus comme des blocs de 1x1 (stockés dans la LIP dans SPIHT). Cette liste est triée des blocs de plus petite taille aux blocs de plus grande taille. Lors du déroulement de l'algorithme, un test de signifiance est réalisé sur chaque ensemble de la LIS à tour de rôle. Si l'ensemble est signifiant et non réduit à un seul coefficient, il est retiré de la liste et partitionné récursivement en quatre sous blocs sur lesquels ce test est effectué à nouveau. Si le bloc est réduit à un seul coefficient significatif, celui ci est ajouté à la LSP. Dans tous les autres cas, l'ensemble est laissé ou ajouté dans la LIS. Les autres coefficients appartenant à $\mathcal{I}$ sont traités ensuite. Si cet ensemble est signifiant, il est séparé en trois blocs de coefficients correspondants aux sous-bandes de plus basses fréquences, et en un nouvel ensemble $\mathcal{I}$ contenant le reste des coefficients. Ces trois nouveaux blocs sont traités comme précédemment. Ce processus de séparation de l'ensemble $\mathcal{I}$ est répété jusqu'à ce qu'il soit insignifiant. Le codage des bits de raffinement est par ailleurs identique à SPIHT.

Cet algorithme regroupe plusieurs idées développées précédemment. Tout d'abord, le partitionnement adaptatif en quad-tree, développée dans [93] et réalisé lors de la séparation de $\mathcal{S}$, permet de repérer rapidement les régions hautement énergétiques et de les coder avec un nombre minimum d'information de signifiance. La séparation récursive en sous-bandes réalisée sur l'ensemble $\mathcal{I}$ et initialement introduite dans [94] permet d'en exploiter la structure hiérarchique en s'intéressant d'abord aux sous-bandes de plus haute énergie a priori. Combinées avec le codage du partitionnement proposé dans SPIHT, ces techniques donnent un codeur par blocs offrant des résultats similaires.

Le codage entropique est uniquement effectué sur les bits de signifiance dus à la séparation d'un ensemble de type $\mathcal{S}$ en quatre sous ensembles. À nouveau, un codeur arithmétique contextuel adaptatif est utilisé. Cette fois-ci, les décisions binaires sont codées une par une (l'alphabet restant donc binaire), en utilisant les signifiances des coefficients précédents comme contexte. Ainsi, la signifiance du premier coefficient est codée à l'aide d'un contexte unique, celle du second à l'aide de deux contextes, et celle du troisième à l'aide de quatre contextes. Un traitement particulier est réalisé pour le dernier coefficient. Si les trois autres coefficients sont insignifiants aucun bit n'est transmis car il est certain que ce dernier coefficient est signifiant (autrement l'ensemble n'aurais pas été séparé). Sinon un codage contextuel est réalisé à l'aide des huit contextes formés par la signifiance des trois coefficients précédents.

2.2.6 EBCOT

Le codeur EBCOT [19], adopté pour le standard JPEG2000, fonctionne en deux passes sur des blocs indépendants de taille moyenne (typiquement 32x32 ou 64x64). Ceux-ci sont codés en un flux hautement progressif et les points de troncatures sont enregistrés pour la deuxième passe qui s'occupe de l'optimisation débit-distorsion. L'allocation de débit optimale, correspondant à la troncature des différents flux pour chaque bloc, est calculée pour différents débits cibles et stockée également dans le flux sous forme compressée, en tant que couche de progressivité. Chaque bloc étant indépendant, il est possible de les réordonner de manière à obtenir une progressivité à la fois en qualité et en résolution, ou de décoder uniquement des zones d'intérêt dans l'image. L'information

de progressivité ayant un coût, les meilleures performances sont obtenues lors du codage en simple couche (SL) au dépend de la progressivité en qualité.

Chaque bloc est codé de manière progressive par plans de bits. Ces blocs sont découpés en sous-blocs de taille 16x16. Un quad-tree de signifiance permet d'éliminer rapidement les sous blocs insignifiants à l'intérieur du bloc considéré. Pour les sous-blocs signifiants restants, le codage de la signifiance s'effectue dans l'ordre classique de parcours à l'aide de deux primitives (Fig. 2.7). La première primitive de *longueur* (run-length coding) indique si quatre coefficients consécutifs sont insignifiants. Si ce n'est pas le cas, la position du premier coefficient signifiant est transmise sur deux bits. Cette primitive n'est utilisée que si tous les voisins (au sens du 8-voisinage) des quatre coefficients du segment sont insignifiants et permet principalement de réduire la complexité du codeur. Dans les autres cas, la primitive de codage des zéros (zero coding) est utilisée. Les coefficients sont traités un par un et leur information de signifiance est transmise. À chaque fois qu'un coefficient devient signifiant, son signe est transmis immédiatement. Les coefficients précédemment signifiants sont ignorés par la passe de signifiance et raffinés en transmettant leurs bits de poids faible.

Ces informations sont compressées à l'aide d'un codeur quasi-arithmétique contextuel, appelé codeur MQ [95] [96]. Il se comporte de manière similaire à un codeur arithmétique tout en ayant une complexité moindre. Le codage de longueur n'utilise qu'un seul contexte pour indiquer si tous les coefficients du segment sont nuls. L'information de position, si elle existe, est transmise sans codage entropique. Pour le codage des zéros, un contexte formé en fonction de l'état de signifiance des 8 voisins du coefficient est utilisé. Les 256 états possibles sont résumés en 9 contextes uniquement pour éviter la dilution des statistiques et permettre un apprentissage rapide des lois conditionnelles. Le signe des coefficients est également codé à l'aide de 5 contextes résumant les 81 états possibles des 4 voisins horizontaux et verticaux du coefficient courant (positif, négatifs ou insignifiants). Enfin, l'information de raffinement est également codée à l'aide de 3 contextes dépendant des bits de poids fort précédemment codés pour ce coefficient et de l'état de signifiance des voisins horizontaux et verticaux. Le codage entropique d'EBCOT repose donc au total sur 18 contextes différents.

La seconde passe consiste à créer des couches de progressivité en allouant a posteriori le débit optimal sur chaque bloc pour atteindre un débit cible. Une optimisation débit-distorsion lagrangienne est réalisée à partir des informations de débit et de distorsion récupérées lors du codage des blocs. En fonction du nombre de couches désirées, la pente correspondant au débit voulu pour chaque couche est déterminée, et les flux de chaque bloc sont tronqués au débit correspondant (Fig. 2.8). La pente et les débits par bloc associés sont alors stockés dans un flux binaire décrivant cette couche. Cette information est compressée par un quad-tree, indiquant la présence ou non de chaque bloc dans une couche, de manière à exploiter la dépendance inter échelle des blocs. Il s'agit en quelque sorte d'une information de signifiance sur les blocs, similaire au principe du zerotree. Lorsqu'un bloc apparaît pour la première fois dans une couche de qualité, l'index du bit de poids fort du coefficient de plus forte amplitude dans ce bloc est également transmis.

L'implémentation d'EBCOT dans la norme JPEG2000 diffère légèrement de celle présentée ici, pour des raisons de réduction de complexité. Notons également que cet

algorithme a été étendu aux contenus tridimensionnels [97], pour le codage vidéo par ondelettes 2D+t.

2.2.7 EZBC

Même les algorithmes les plus récents, comme EZBC [98], ne font qu'améliorer l'application des principes de partitionnement d'ensemble insignifiants introduits par EZW et SPIHT. Le principe de codage est similaire à SPECK, l'innovation de ce codeur provenant principalement de l'exploitation de la dépendance entre les noeuds du quad-tree de signifiance. Le partitionnement en quad-tree est de plus réalisé indépendamment dans chaque sous-bande permettant une meilleure séparation des statistiques de signifiance et un apprentissage plus efficace à l'aide de contextes plus étendus.

La première étape de codage consiste à créer un quad tree de signifiance pour chaque sous bande. Les feuilles de cet arbre représentent les coefficients de la sous-bande en question, tandis que les noeuds internes sont initialisés à la valeur d'amplitude maximale de tous leurs descendants. Ainsi le noeud racine correspond à l'amplitude maximale de la sous-bande. Les différentes couches de l'arbre sont codées du noeud racine aux feuilles en testant la signifiance des noeuds. De même que dans SPIHT et SPECK, ce codage s'effectue très rapidement en s'appuyant sur la gestion de listes de coefficients et d'ensembles.

La dépendance entre les noeuds de l'arbre est exploitée lors du codage entropique contextuel. Pour chaque noeud, un contexte est formé à partir de l'état de signifiance de ses huit voisins situés au même niveau de l'arbre, ainsi que du noeud situé dans l'arbre de la sous-bande parente au niveau inférieur (Fig. 2.9). Le fait de considérer le noeud de la sous-bande parente au niveau inférieur tient compte du changement d'échelle entre les sous-bandes et permet d'exploiter la dépendance inter-échelle. Les 512 états possibles sont réduits à 20 contextes comme dans EBCOT, pour améliorer l'apprentissage et l'adaptation des lois associées.

Ce codeur offre des performances comparables à EBCOT, et a également été adapté au codage vidéo sous le nom de MC-EZBC [99]. Nous avons par ailleurs entamé une modification d'EZBC pour améliorer la quantification des coefficients signifiants en considérant des lois gaussiennes généralisées. L'information de raffinement est alors quantifiée et codée d'après ce modèle, de manière similaire à EQ, plutôt que d'utiliser un codeur arithmétique adaptatif et une quantification uniforme. Cette modification offre toutefois pour l'instant des performances similaires à la version originale d'EZBC.

Les tableaux 2.2 et 2.3 présentent le débit utilisé pour le codage des différents types d'information. Il y apparaît clairement qu'un codage efficace de l'information de signifiance est primordial. De plus, les choix fait dans SPIHT et SPECK de ne pas compresser l'information de signe et de raffinement sont justifiés par l'impact relativement faible de leur codage entropique sur le débit total.

Si l'on considère l'introduction d'erreurs binaires dans le flux compressé[1], l'information de signifiance est très fragile dans le sens ou une seule erreur peut avoir un

[1] dues à sa transmission sur un canal bruité, au vieillissement du support de stockage, ou encore à l'interaction des mémoires avec les rayonnements interstellaires

type d'information	débit (bits) 16684	33427	65850	133255	268896
signifiance	12040 (72%)	23491 (70%)	45066 (68%)	90558 (68%)	180524 (67%)
signe	2514 (15%)	5172 (15%)	9983 (15%)	20186 (15%)	43193 (16%)
raffinement	2130 (13%)	4764 (14%)	10801 (16%)	22511 (17%)	45179 (17%)

TAB. 2.2: Répartition du débit entre les différentes informations par le codeur EZBC pour l'image *lena* 512x512 décomposée sur 5 niveaux d'ondelettes séparables. Les débits cibles correspondent approximativement à des taux de compression de 128 :1, 64 :1, 32 :1, 16 :1 et 8 :1.

type d'information	non codée (bits)	compressée (bits)	taux
signifiance	195763	180524	92.22 %
signe	48010	43193	89.97 %
raffinement	47788	45179	94.54 %
total	291561	268896	92.23 %

TAB. 2.3: Taux de compression des différentes informations du codeur EZBC pour l'image *lena* 512x512 décomposée sur 5 niveaux d'ondelettes séparables codée à 1 bpp.

impact catastrophique sur la qualité de l'image décodée. Les informations de signe et de raffinement en revanche, en supposant qu'elles ne soient pas codées entropiquement, sont robustes car elles ne concernent qu'un seul coefficient. Le tableau 2.2 suggère que dans les schémas de codage présentés, environ deux tiers de l'information compressée est fragile, ce qui motive la recherche de moyens de les coder de manière robuste en codage conjoint source-canal.

2.2.8 Comparaison des différents codeurs

Le succès des approches par ondelettes en compression s'explique en grande partie du fait de l'apparition de codeurs de sous-bandes efficaces. Le codeur EZW est le premier à avoir fourni des performances débit-distorsion remarquables tout en permettant un décodage progressif de l'image. Le principe des zerotrees, ou d'autres structures de partitionnement en ensembles de zéros, permet en effet de tenir compte de dépendance résiduelle des coefficients entre eux. Plus précisément, les coefficients de haute énergie étant groupés spatialement, leur position est codée efficacement par complémentarité en indiquant la position des ensembles de coefficients peu énergétiques. Après séparation des coefficients peu énergétiques et des coefficients énergétiques, ces derniers sont relativement indépendants et peuvent être codés efficacement à l'aide de techniques de quantification et de codage entropique simples.

Les codeurs successifs comme SPIHT et SPECK ont amélioré le principe de EZW en proposant un codage plus efficace de l'information de signifiance qui code la position

des coefficients énergétiques. Comme cette information représente une grande part du débit de l'image, son codage efficace est primordial. Ce même principe est encore utilisé dans les codeurs les plus récents que sont EBCOT et EZBC, offrant des performances supérieures à SPIHT du fait de l'utilisation de contextes de codages de plus grande dimension, au prix d'une complexité accrue. Notons cependant que l'information de signifiance est fragile et qu'il est essentiel, dans un environnement bruité, de la protéger pour obtenir une image décodée correcte.

L'ensemble de ces codeurs permettent un décodage progressif de l'image en tronquant le flux binaire la représentant, contrairement à SFQ et EQ. Ces derniers permettent cependant d'obtenir des performances équivalentes ou légèrement supérieures à EBCOT et EZBC pour une complexité moindre. Parmi les codeurs progressifs, les codeurs EZW, SPIHT et SPECK supposent que l'énergie des coefficients d'ondelettes décroît des échelles grossières aux échelles fines, et que les coefficients signifiants ont probablement des fils signifiants. Les coefficients sont codés dans un ordre de parcours défini par l'algorithme et fixe pour toutes les images. En revanche, les codeurs EZBC et EBCOT permettent de créer des flux binaires progressifs indépendants. En particulier, le codeur EBCOT permet de coder chaque bloc de 32x32 coefficients indépendemment. En enregistrant les points de troncature de chaque flux binaire et les distorsions correspondantes, une optimisation débit-distorsion est alors réalisable a posteriori. Ainsi, il est à la fois possible d'obtenir une représentation non progressive de l'image en ne réalisant l'allocation de débit que pour un seul débit de codage (EBCOT SL), et une représentation progressive en réalisant l'allocation de débit pour plusieurs débits cibles. Cette information sur la répartition du débit entre les différents blocs doit toutefois être représentée dans le flux binaire final, ce qui explique les performances légèrement inférieures du mode de codage progressif (EBCOT GS).

Le tableau 2.4 compare les différents codeurs en termes de performance pour différents débits. Le codeur non progressif offrant les meilleures performances est EQ, bien que nous n'ayons pas connaissance de son application dans un format de compression d'image. Les codeurs progressifs les plus performants sont EZBC et EBCOT. Le premier a surtout été utilisé à l'heure actuelle dans le cadre de la compression vidéo, tandis que le second est utilisé dans la norme JPEG2000.

Image	Codeur	débit (bpp)				
		0.0625	0.125	0.25	0.5	1.0
lena	EZW	27.54	30.23	33.17	36.28	39.55
	SPIHT	28.38	31.10	34.11	37.21	40.44
	SPECK	-	-	34.03	37.10	40.25
	EBCOT GS*	28.10	31.05	34.16	37.29	40.48
	EZBC	-	-	34.35	37.47	40.62
	SFQ	-	-	34.33	37.36	40.52
	EQ	-	-	**34.57**	**37.68**	**40.88**
	EBCOT SL	**28.30**	**31.22**	34.28	37.43	40.61
barbara	EZW	23.10	24.03	26.77	30.53	35.14
	SPIHT	23.35	24.86	27.58	31.40	36.41
	SPECK	-	-	27.76	31.54	36.49
	EBCOT GS*	23.34	25.37	28.40	32.29	37.11
	EZBC	-	-	28.25	32.15	37.28
	SFQ	-	-	28.29	32.15	37.03
	EBCOT SL	**23.45**	**25.55**	**28.55**	**32.48**	**37.37**
goldhill	EZW	-	-	30.31	32.87	36.20
	SPIHT	-	-	30.56	33.13	36.55
	SPECK	-	-	30.50	33.03	36.36
	EBCOT GS*	-	-	30.59	33.25	36.59
	EZBC	-	-	30.74	33.47	36.90
	SFQ	-	-	30.71	33.37	36.70
	EQ	-	-	**30.76**	**33.42**	**36.96**
	EBCOT SL	-	-	30.71	33.35	36.72
cafe	SPIHT	18.92	20.64	22.99	26.45	31.70
	SPECK	18.93	20.61	22.87	26.31	31.47
	EBCOT GS*	19.06	20.82	23.20	26.87	32.03
	EZBC	**19.11**	20.87	**23.32**	**27.00**	**32.43**
	EBCOT SL	19.10	**20.88**	23.29	**27.00**	32.27
bike	SPIHT	23.36	25.79	29.04	32.94	37.66
	SPECK	23.31	25.59	28.84	32.69	37.33
	EBCOT GS*	23.78	26.37	29.60	33.46	38.09
	EZBC	23.75	26.11	29.58	33.53	38.24
	EBCOT SL	**23.88**	**26.49**	**29.76**	**33.68**	**38.29**

$\star$: mode progressif à granularité fine contenant 50 couches de progressivité.

TAB. 2.4: Tableau comparatif des performances débit-distorsion en termes de PSNR des codeurs de sous-bandes. Une décomposition en ondelettes séparables sur 4 niveaux est utilisée.

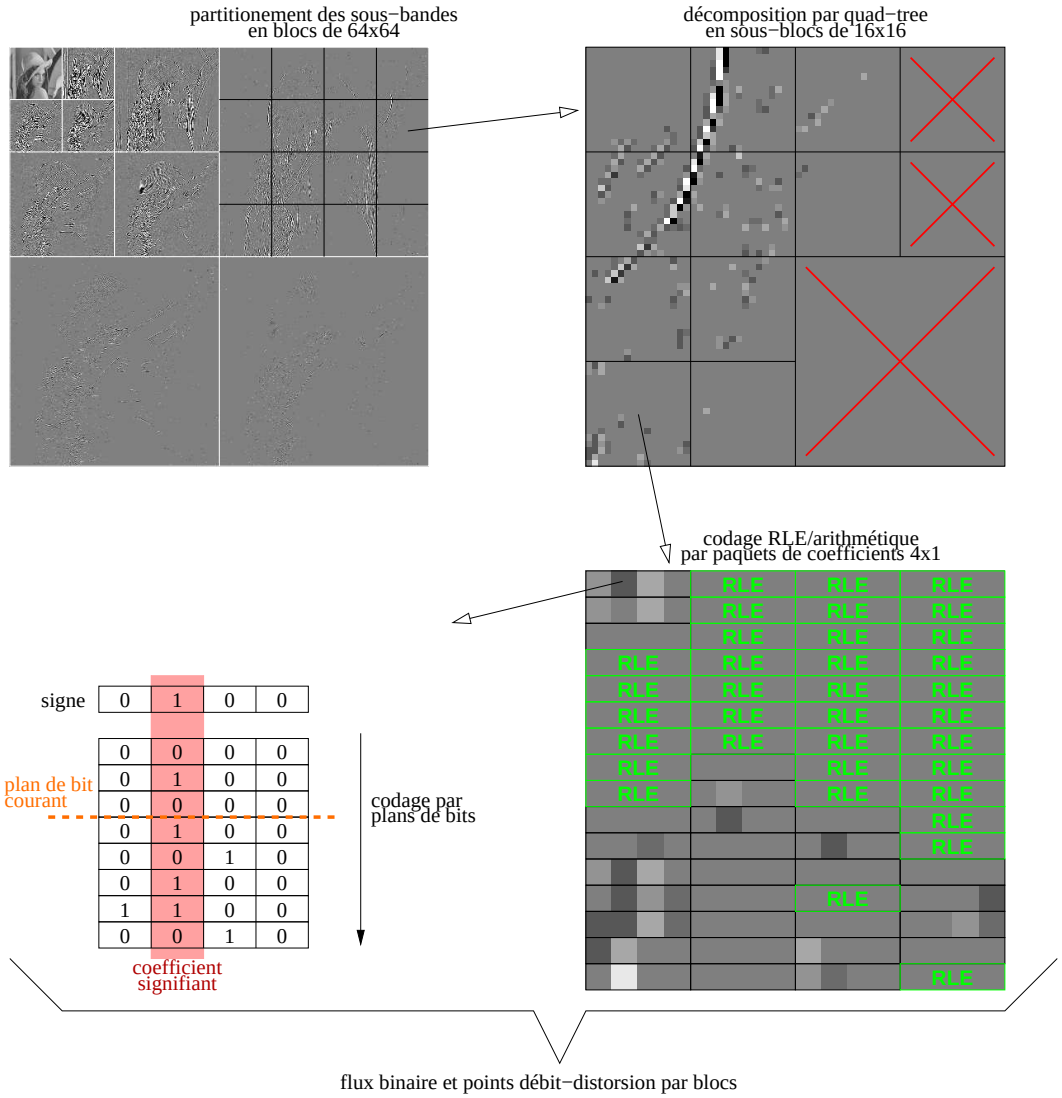

FIG. 2.7: Codage des sous-bandes d'ondelettes séparables par EBCOT. Chaque sous-bande est traitée indépendamment en la séparant tout d'abord en blocs de taille 64x64. Un quad-tree de signifiance élimine rapidement les sous-blocs 16x16 insignifiants de chaque blocs. Les sous-blocs signifiant sont ensuite codés par plans de bits à l'aide de deux primitives. Cette passe de codage fournit un flux binaire encodé et les valeurs de distorsion pour chaque débit.

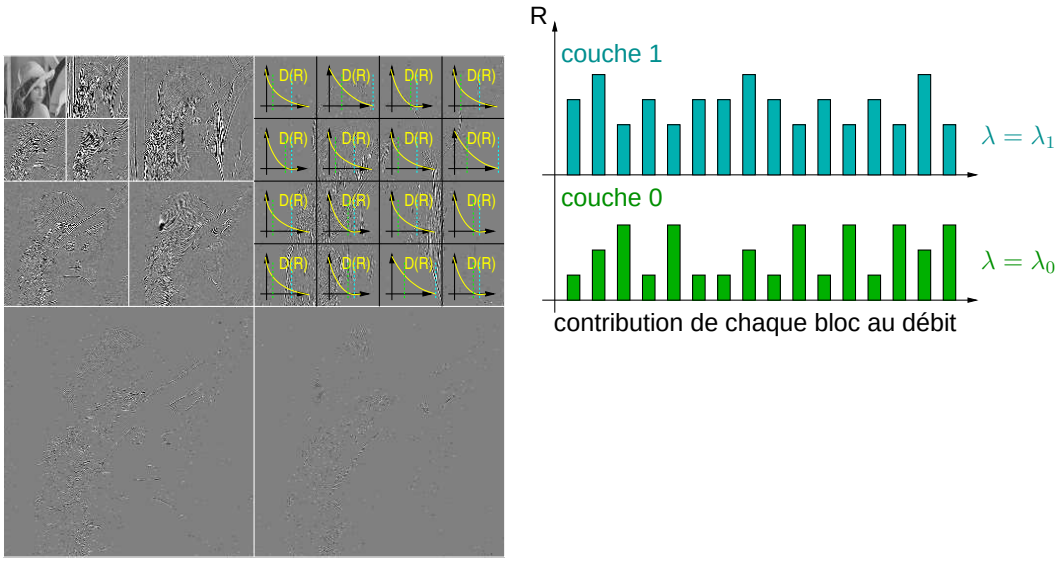

FIG. 2.8: Optimisation débit-distorsion a posteriori par le codeur EBCOT. Un certain nombre de couches de progressivité sont construites en tronquant les flux issus du codage de chaque bloc au débit correspondant à une pente commune sur les courbes débit-distorsion de chaque bloc.

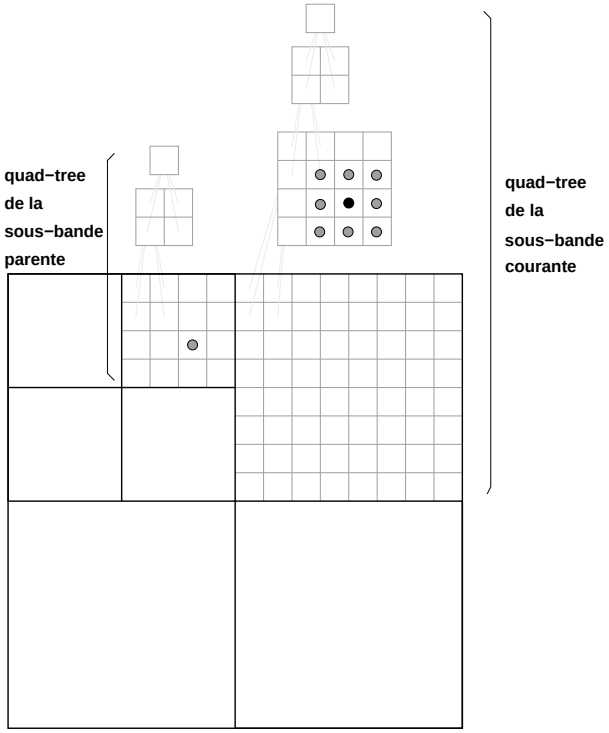

FIG. 2.9: Contexte pour le codage de la signifiance des noeuds dans EZBC. Le noeud courant est représenté en noir et son contexte intra et inter sous-bande en gris.

Chapitre 3

Compression par contourlettes

Les transformées redondantes ont été utilisées dans divers domaines du traitement du signal, comme le débruitage, du fait de leur flexibilité accrue en comparaison des transformées à échantillonage critique. Cependant, elle ne semblent pas être un choix naturel pour la compression, car l'espace de sortie est plus grand que l'espace d'entrée. Néanmoins, dans le contexte de l'approximation non-linéaire, un nombre restreint de coefficients d'une transformée redondante peut représenter le signal plus fidèlement qu'une transformée non redondante, du fait de la plus grande liberté de choix de l'ensemble des atomes de codage. Il en résulte que les transformées redondantes directionnelles fournissent de bonnes approximations non linéaires à bas débit, tout en étant moins efficaces que les transformées non redondantes à haut débit.

Lorsque l'ensemble de fonctions de base est arbitraire, il est possible d'extraire une représentation creuse de l'image en appliquant l'algorithme de Matching pursuit [69]. Cette technique reste cependant coûteuse en temps de calcul et sous-optimale. En introduisant une certaine structure à l'ensemble des atomes de codages, des algorithmes plus efficaces utilisant des transformées linéaires sont conçus en s'appuyant sur la théorie des frames, tout en conservant une plus grande flexibilité par rapport aux transformées à échantillonnage critique.

La transformée en contourlettes [44] fournit une analyse multirésolution, avec un nombre arbitraire de sous-bandes directionnelles à chaque niveau, et forme une frame étroite de redondance faible. En utilisant la théorie des projections sur espaces convexes, Kingsbury et al. proposent dans [56] un algorithme itératif pour minimiser la distorsion introduite par la quantification dans le domaine transformé de la transformée en ondelettes complexe, également redondante.

Ici, nous proposons un schéma hybride dans lequel les basses fréquences sont codées en utilisant la transformée en ondelettes séparables, tandis que les hautes fréquences sont codées en utilisant la transformée en contourlettes. La redondance résiduelle est ensuite exploitée en utilisant la technique de projection sur ensembles convexes pour obtenir un gain significatif par rapport à la transformée en ondelettes séparables à très bas débit. Cette transformée hybride a également l'avantage de fournir une analyse directionnelle fine des hautes fréquences, pour une complexité similaire à la transformée

en ondelettes séparables.

3.1 Représentation en contourlettes

Introduite dans [44], la transformée en contourlettes fournit une analyse multirésolution et directionnelle d'un signal bidimensionnel. La décomposition multirésolution est effectuée au moyen d'une pyramide laplacienne redondante. L'analyse directionnelle s'effectue ensuite au moyen d'un banc de filtres directionnels à échantillonage critique appliqué à chaque sous-bande de la pyramide laplacienne excepté la sous-bande de basse fréquence (Fig. 3.1).

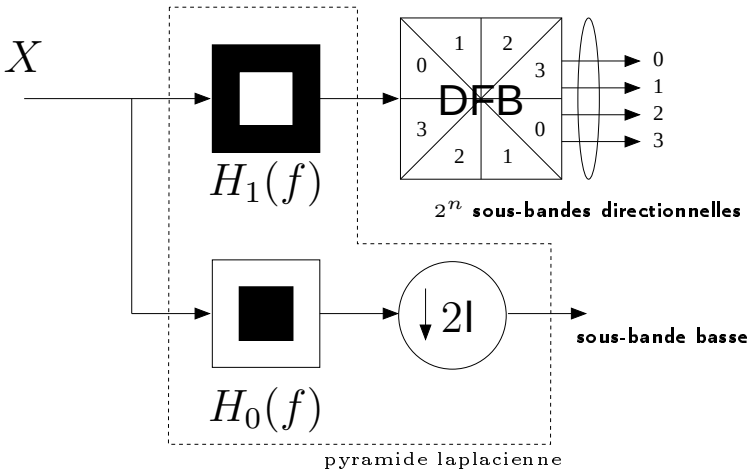

FIG. 3.1: Un niveau d'analyse la tranformée en contourlette. Les régions noires indiquent la réponse fréquentielle idéale du filtre. La sous-bande de haute fréquence obtenue après filtrage par H_1 n'est pas sous-échantillonnée.

3.1.1 Pyramide laplacienne

Une pyramide laplacienne [47] est une représentation multirésolution redondante obtenue en décomposant le signal en une approximation de basse résolution et l'erreur de reconstruction correspondant à chaque échelle (Fig. 3.2). Cette décomposition est réalisée de manière itérative. Un filtre passe-bas d'analyse H_0 est appliqué sur le signal puis celui-ci est sous-échantillonné par M pour en obtenir une représentation basse résolution $\mathsf{y}_{\mathcal{L}}$. Ce signal d'approximation est alors sur-échantillonné par M puis filtré par le filtre passe-bas de synthèse $G0$ pour obtenir un signal dégradé de même résolution que le signal d'origine. L'erreur haute fréquence $\mathsf{y}_{\mathcal{H}}$ introduite par les filtrages et les ré-échantillonnages est alors calculée en soustrayant le signal dégradé au signal d'origine. On obtient alors après cette étape d'analyse un signal d'approximation $\mathsf{y}_{\mathcal{L}}$ de résolution inférieure et un signal d'erreur $\mathsf{y}_{\mathcal{H}}$ de résolution identique au signal d'origine. Le facteur

de redondance de chaque étape d'analyse est alors donné par $1 + |\det \mathsf{M}|^{-1}$. Ce processus d'analyse est itéré sur le signal d'approximation $\mathsf{y}_{\mathcal{L}}$ pour obtenir la représentation pyramidale. La redondance totale est donc bornée par :

$$\sum_{i=0}^{\infty} |\det \mathsf{M}|^{-n} = \frac{1}{1 - |\det \mathsf{M}|^{-1}} = \frac{|\det \mathsf{M}|}{|\det \mathsf{M}| - 1}.$$

Dans le cadre de la décomposition en pyramide laplacienne des images, il est usuel de choisir un facteur de sous-échantillonnage de 2 verticalement et horizontalement ($\mathsf{M} = 2\mathsf{I}$). Cette structure d'échantillonnage a également été choisie pour la transformée en contourlettes et la redondance totale est alors bornée par $\frac{4}{3}$. Les filtres passe-bas H_0 et G_0 ont également été choisis pour correspondre aux filtres passe-bas respectivement d'analyse et de synthèse des ondelettes 9/7 séparables [12].

L'opérateur d'analyse A se décrit formellement en notant H l'opérateur de filtrage par H_0 suivi du sous-échantillonnage par M et G l'opérateur de sur-échantillonnage par M suivi du filtrage par G_0. L'opération d'analyse s'écrit alors :

$$\begin{pmatrix} \mathsf{y}_{\mathcal{L}} \\ \mathsf{y}_{\mathcal{H}} \end{pmatrix} = \underbrace{\begin{pmatrix} \mathsf{H} \\ \mathsf{I} - \mathsf{GH} \end{pmatrix}}_{\mathsf{A}} \mathsf{x}.$$

L'opération usuelle de synthèse S, décrite dans [47], consiste simplement à ajouter le signal d'erreur $\mathsf{y}_{\mathcal{H}}$ au signal $\mathsf{y}_{\mathcal{L}}$ une fois ce dernier sur-échantillonné et filtré par $G0$. En d'autres termes, l'opération usuelle de synthèse s'écrit :

$$\mathsf{x} = \underbrace{\begin{pmatrix} \mathsf{G} & \mathsf{I} \end{pmatrix}}_{\mathsf{S}} \begin{pmatrix} \mathsf{y}_{\mathcal{L}} \\ \mathsf{y}_{\mathcal{H}} \end{pmatrix}.$$

Cet opérateur permet bien de reconstruire le signal d'origine, car $\mathsf{SA} = \mathsf{I}$. L'opération d'analyse étant redondante, il existe cependant plusieurs opérateurs de synthèse capables de reconstruire le signal original x. Parmi l'ensemble de ces opérateurs, l'opérateur $\mathsf{A}^{\perp}$ pseudo-inverse de l'opérateur d'analyse minimise l'erreur de reconstruction en présence de bruit blanc additif dans le domaine transformé. Dans le cas général où les filtres G et H sont quelconques, les filtres de synthèse correspondant à cet opérateur s'obtiennent en inversant la matrice de polynômes de Laurent correspondante dans le domaine polyphase [4]. Toutefois, si G et H sont à réponse impulsionnelle finie, rien

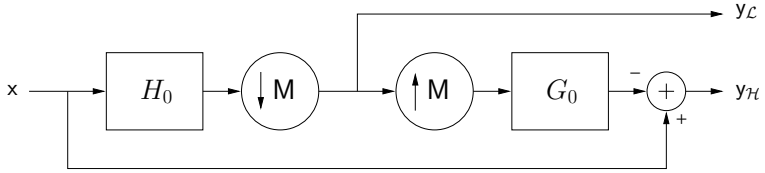

FIG. 3.2: Étape d'analyse de la pyramide laplacienne.

ne garantit que l'opération inverse s'obtienne par filtrage à réponse impulsionnelle finie également (l'inversion d'une matrice de polynômes faisant apparaître des fractions rationnelles). Ainsi dans le cas général, les filtres correspondant à l'opérateur pseudo inverse sont à réponse impulsionnelle infinie, ce qui peut éventuellement être problématique pour l'implémentation. En revanche, dans le cas où G et H sont orthogonaux par rapport à la matrice de sous-échantillonnage M, c'est à dire

$$\forall \mathsf{n} \in \mathbb{Z}^2, \sum_{\mathsf{k}\in\mathbb{Z}^2} g_0[\mathsf{k}]g_0[\mathsf{k}-\mathsf{Mn}] = 0 \text{ et } h_0[\mathsf{n}] = g_0[-\mathsf{n}],$$

on a $\mathsf{G} = \mathsf{H}^\top$. L'opérateur pseudo-inverse s'obtient alors par simple transposition de l'opérateur d'analyse :

$$\mathsf{A}^\top = \begin{pmatrix} \mathsf{G} & \mathsf{I}-\mathsf{GH} \end{pmatrix}.$$

Il est donc aisé d'effectuer la reconstruction pseudo-inverse du signal par filtrage à réponse impulsionnelle finie selon la structure représentée figure 3.3. Dans le cas des filtres 9/7, cette propriété d'orthogonalité est quasiment vérifiée et l'opérateur de synthèse transposé de l'opérateur d'analyse a donc été choisi plutôt que l'opérateur usuel pour la reconstruction de la pyramide dans le cas des contourlettes. Notons que bien que nous reprenions ici le formalisme utilisé dans [4], il avait déjà été remarqué lors de l'élaboration de la pyramide orientable [65] que l'opérateur de synthèse usuel [47] n'est pas optimal au sens de la théorie de frames.

3.1.2 Analyse directionnelle

La décomposition directionnelle est obtenue après n itérations d'un banc de filtres en éventail à deux canaux, avec rééchantillonage préalable pour obtenir les 2^n sous-bandes orientées (Fig. 3.6).

3.1.2.1 Prototypes de filtres en éventail

L'analyse directionnelle repose sur une paire de filtres en éventail biorthogonaux. Ces filtres sont obtenus à partir d'une paire de filtres biorthogonaux unidimensionnels qui sont transformés algébriquement en une paire de filtres quinconces et quinconces

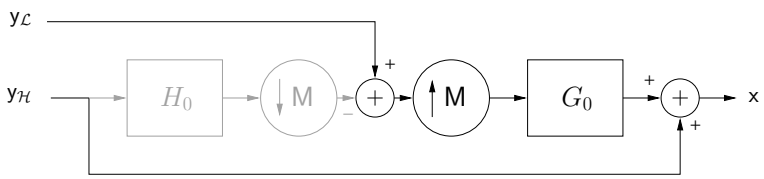

FIG. 3.3: Étape de synthèse de la pyramide laplacienne. La reconstruction usuelle est représentée en noir. La reconstruction correspondant à l'opérateur transposé s'obtient en y ajoutant le filtrage par H_0 et le sous-échantillonnage par M représentés en gris.

complémentaires [7]. La paire de filtres en éventail s'obtient ensuite à partir de la paire de filtres quinconce par modulation, soit selon l'axe vertical, soit selon l'axe horizontal.

La transformation des filtres unidimensionnels en filtres bidimensionnels quinconces est similaire à celle de McClellan [6] excepté qu'elle s'effectue dans le domaine polyphase. Le fait d'utiliser des paires de filtres bidimensionnels quinconces ou en éventail permet d'implémenter le banc de filtres de manière séparable dans le domaine polyphase, bien que les filtres eux-mêmes ne soient pas séparables. Ceci apporte un gain en complexité non négligeable car le filtrage s'effectue en temps linéaire, comme pour les filtres séparables, au lieu d'un temps quadratique par rapport à la taille du filtre.

3.1.2.2 Banc de filtres directionnels

Cette paire de filtres en éventail $(F0, F1)$ sert à l'élaboration récursive du banc de filtres d'analyse directionnelle. Un premier filtrage sépare le signal en une composante en éventail verticale et une composante en éventail horizontale. Il est alors possible de sous-échantillonner de manière critique ce signal sur une lattice quinconce définie par la matrice :

$$\mathsf{Q}_0 = \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}.$$

En effet, la paire de filtres étant duale, il est possible de reconstruire le signal parfaitement après sur-échantillonage par $\mathsf{Q_0}$ ou $\mathsf{Q_1}$ en utilisant les filtres de synthèse $(G0, G1)$ correspondants, obtenus également par simple modulation des filtres d'analyse.

L'étape suivante de décomposition directionnelle applique à nouveau à chaque sortie le banc de filtres en éventail puis sous-échantillonne par la matrice :

$$\mathsf{Q}_1 = \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}.$$

Cette matrice Q_1 définit également une lattice quinconce, mais le fait de l'utiliser plutôt que Q_0 pour le second sous-échantillonnage permet d'obtenir quatre sous-bandes orientées de manière identique à l'image originale, car $\mathsf{Q}_0\mathsf{Q}_1 = 2\mathsf{I}$.

Le signal sur lequel ce banc de filtres est appliqué ayant déjà été sous-échantillonné, le filtrage équivalent avant sous-échantillonnage s'obtient en utilisant l'identité remarquable des bancs de filtres (Fig. 1.9), et correspond à un filtrage par une paire de filtres en damier (Fig. 3.4).

Les étapes suivantes s'effectuent de manière légèrement différente en appliquant tout d'abord un ré-échantillonnage du signal en sortie de chaque branche. Ce ré-échantillonnage est effectué en sous-échantillonnant[1] le signal par l'une des matrices suivantes :

[1] Bien qu'il s'agisse d'un ré-échantillonnage le terme de sous-échantillonage ou sur-échantillonage est utilisé pour distinguer le sens dans lequel est appliquée l'opération.

$$\mathsf{R}_0 = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}, \mathsf{R}_1 = \begin{pmatrix} 1 & -1 \\ 1 & 0 \end{pmatrix}, \tag{3.1}$$

$$\mathsf{R}_2 = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}, \mathsf{R}_3 = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix}. \tag{3.2}$$

Ces matrices, appelées matrices unimodulaires, sont toutes de déterminant égal à 1. Ainsi le ré-échantillonnage est totalement inversible et sans pertes. A chaque étape, pour la première moitié du banc de filtres, les branches paires sont sous-échantillonnées par R_0 tandis que les branches impaires sont sous-échantillonnées par R_1. De même, pour la seconde moitié du banc de filtres, les branches paires sont sous-échantillonnées par R_2 tandis que les branches impaires sont sous-échantillonnées par R_3.

Le banc de filtres en éventail est appliqué par la suite sur chaque branche. Ceci permet d'obtenir des filtres équivalents en parallélogramme (Fig. 3.5). Hormis les deux premières étapes d'analyse directionnelles détaillées précédemment, toutes les autres étapes s'effectuent de la même manière jusqu'à obtention du nombre de sous-bandes désirées, correspondant nécessairement à une puissance de 2. Le signal obtenu en sortie de chaque sous-bande est alors défini sur un parallélogramme et est ré-échantillonné une dernière fois sur un rectangle pour une représentation plus aisée. Ainsi, un signal de taille $w \times h$ est décomposée en 2^n sous-bandes directionnelles, dont les 2^{n-1} correspondant aux orientations plutôt verticales ($[\frac{\pi}{4}, \frac{3\pi}{4}]$) sont de taille $\frac{w}{2^{n-1}} \times \frac{h}{2}$, et les 2^{n-1} correspondant aux orientations plutôt horizontales ($[-\frac{\pi}{4}, \frac{\pi}{4}]$) sont de taille $\frac{w}{2} \times \frac{h}{2^{n-1}}$ (Fig. 3.8).

Exemple 8: Nous allons illustrer le principe d'élaboration du banc de filtres directionnels à l'aide des filtres 5/3. Ceux ci sont définis par leur réponse impulsionnelle, représentée ici sous forme matricielle :

$$\mathsf{H}_0 = \frac{\sqrt{2}}{8} \begin{pmatrix} -1 & 2 & 6 & 2 & -1 \end{pmatrix}, \tag{3.3}$$

$$\mathsf{H}_1 = \frac{1}{2\sqrt{2}} \begin{pmatrix} -1 & 2 & -1 \end{pmatrix}. \tag{3.4}$$

$$\tag{3.5}$$

En appliquant la transformation [7], on obtient les filtres quinconces suivants :

FIG. 3.4: Transformation des filtres en éventail en filtres en damier. La paire de filtres en éventail (F_0, F_1) devient une paire de filtres en damier (D_0, D_1) après inversion du filtrage et du sous-échantillonage par Q_0.

$$\mathsf{Q}_0 = \frac{\sqrt{2}}{32}\begin{pmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -2 & 4 & -2 & 0 \\ -1 & 4 & 28 & 4 & -1 \\ 0 & -2 & 4 & -2 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{pmatrix}, \tag{3.6}$$

$$\mathsf{Q}_1 = \frac{1}{4\sqrt{2}}\begin{pmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{pmatrix}. \tag{3.7}$$

Les filtres en éventail s'obtiennent aisément en multipliant les vecteurs lignes par -1 *toutes les deux lignes, ce qui a pour effet de translater la réponse fréquentielle du filtre de* 1 *dans la direction verticale :*

$$\mathsf{F}_0 = \frac{\sqrt{2}}{32}\begin{pmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & 2 & -4 & 2 & 0 \\ -1 & 4 & 28 & 4 & -1 \\ 0 & 2 & -4 & 2 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{pmatrix}, \tag{3.8}$$

$$\mathsf{F}_1 = \frac{1}{4\sqrt{2}}\begin{pmatrix} 0 & 1 & 0 \\ -1 & 4 & -1 \\ 0 & 1 & 0 \end{pmatrix}. \tag{3.9}$$

En appliquant l'identité remarquable des bancs de filtres (Fig. 1.9) pour intervertir le sous-échantillonnage par Q *et le filtrage par* (F_0, F_1)*, on obtient la paire de filtres en damier suivants :*

FIG. 3.5: Transformation des filtres en éventail en filtres en parallélogramme. La paire de filtres en éventail (F_0, F_1) devient une paire de filtres en parallélogramme (P_0, P_1) après inversion du filtrage et du sous-échantillonage par R_0.

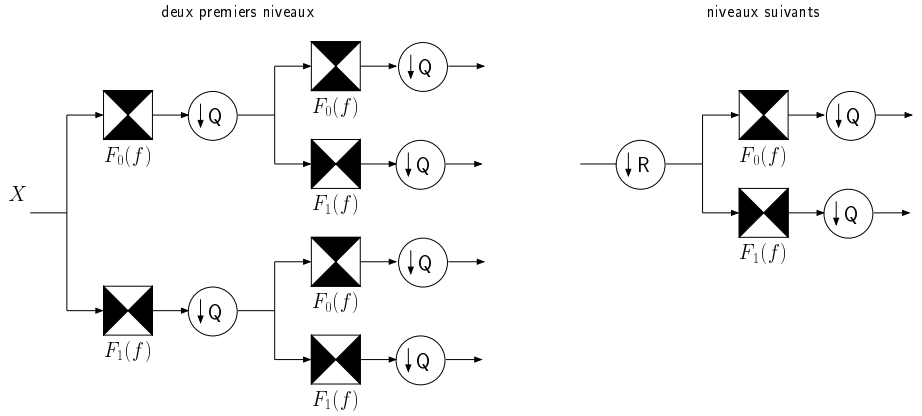

FIG. 3.6: Banc de filtres d'analyse directionnel (DFB). Les réponses idéales des filtres sont représentées en noir. À chaque étape, le signal est sous-échantillonné de manière critique d'un facteur 2 sur la lattice quinconce générée par Q. $F0$ et $F1$ sont une paire de filtres biorthogonaux. R est une matrice unimodulaire de ré-échantillonage.

$$\mathsf{D}_0 = \frac{\sqrt{2}}{32}\begin{pmatrix} -1 & 0 & 2 & 0 & -1 \\ 0 & -4 & 0 & 4 & 0 \\ 2 & 0 & 28 & 0 & 2 \\ 0 & 4 & 0 & -4 & 0 \\ -1 & 0 & 2 & 0 & -1 \end{pmatrix}, \tag{3.10}$$

$$\mathsf{D}_1 = \frac{1}{4\sqrt{2}}\begin{pmatrix} 1 & 0 & -1 \\ 0 & 4 & 0 \\ -1 & 0 & 1 \end{pmatrix}. \tag{3.11}$$

De même il est possible d'obtenir des filtres en parallélogramme en effectuant un ré-échantillonage préalable du signal, par exemple par la matrice de ré-échantillonnage R_0. *En appliquant à nouveau l'identité remarquable des bancs de filtres on remarque qu'un sous-échantillonnage par* R_0 *suivi du filtrage par* (F_0, F_1) *est équivalent au filtrage par* (P_0, P_1) *suivi du même sous-échantillonnage par* R_0, *où les filtres* (P_0, P_1) *sont définis par :*

$$\mathsf{P}_0 = \frac{\sqrt{2}}{32}\begin{pmatrix} 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & -4 & 4 & 0 \\ 0 & 2 & 28 & 2 & 0 \\ 0 & 4 & -4 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \end{pmatrix}, \tag{3.12}$$

$$\mathsf{P}_1 = \frac{1}{4\sqrt{2}}\begin{pmatrix} 0 & 1 & -1 \\ 0 & 4 & 0 \\ -1 & 1 & 0 \end{pmatrix}. \tag{3.13}$$

3.2 Étude des filtres de contourlettes

Dans le cadre de la compression, la propriété d'orthogonalité des filtres de la transformée directionnelle et des filtres de la pyramide laplacienne est préférable pour l'allocation du débit dans le domaine transformé. Cependant, cette propriété est antagoniste avec les propriétés de phase linéaire et de reconstruction parfaite essentielles à la qualité des images restituées. Ainsi, les filtres utilisés pour la décomposition en contourlettes sont des filtres biorthogonaux symétriques, tout en cherchant à s'approcher au mieux de la condition d'orthogonalité.

3.2.1 Décomposition pyramidale

Pour la décomposition pyramidale, nous avons considéré trois filtres biorthogonaux différents, à savoir les filtres 9/7 [12], 13/11 [100] et MPEG [101]. Ces filtres sont définis par les valeurs suivantes :

filtre	0	±1	±2	±3	±4	±5	±6
$H_0^{9/7}$	0.85270	0.37740	−0.11062	−0.02385	0.03783		
$G_0^{9/7}$	0.78849	0.41809	−0.04069	−0.06454			
$H_0^{13/11}$	0.76725	0.38327	−0.06888	−0.03348	0.04728	0.00376	−0.00847
$G_0^{13/11}$	0.83285	0.44811	−0.06916	−0.10874	0.00629	0.01418	
H_0^{MPEG}	0.57452	0.41984	0.11049	−0.06629	−0.08839	0.00000	0.04419
G_0^{MPEG}	0.70711	0.44194	0.00000	−0.11049	0.00000	0.02210	

La figure 3.9 représente l'image reconstruite à partir d'une sous-bande de l'image *zoneplate*. On y constate que le filtre MPEG offre une bien meilleure sélectivité que les autres filtres biorthogonaux 9/7 et 13/11. Ceci réduit également les problèmes de recouvrement de spectre perceptibles pour les autres filtres. Cependant, ce filtre est loin de respecter la propriété d'orthogonalité. En effet, les tableaux 3.2 et 3.3 évaluent le produit scalaire intra et inter échelle entre différentes fonctions de base issues du même filtre.

Alors que la corrélation entre les fonctions de base à deux niveaux successifs ne dépasse par 5% de leur norme pour les filtres 9/7 et 13/11, elle atteint les 20% pour le filtre MPEG. De même, pour deux coefficients voisins, les fonctions de base issues des filtres 9/7, 13/11 et MPEG sont corrélées respectivement à 1%, 6% et 10%. Ainsi il n'est pas justifié d'approximer l'opérateur de synthèse pseudo-inverse par l'opérateur transposé de l'opérateur d'analyse dans le cas des filtres MPEG. Il serait possible d'utiliser l'opérateur traditionnel de reconstruction par simple addition pour garantir la reconstruction parfaite lors d'un codage sans pertes, mais cet opérateur introduit une distorsion supplémentaire non négligeable par rapport à l'opérateur pseudo-inverse en présence de bruit. Enfin, il serait également possible d'effectuer la reconstruction pseudo-inverse en construisant le banc de filtres à réponse impulsionnelle infinie correspondant, bien que cette voie n'ait pas été poursuivie dans cette thèse. Notons toutefois

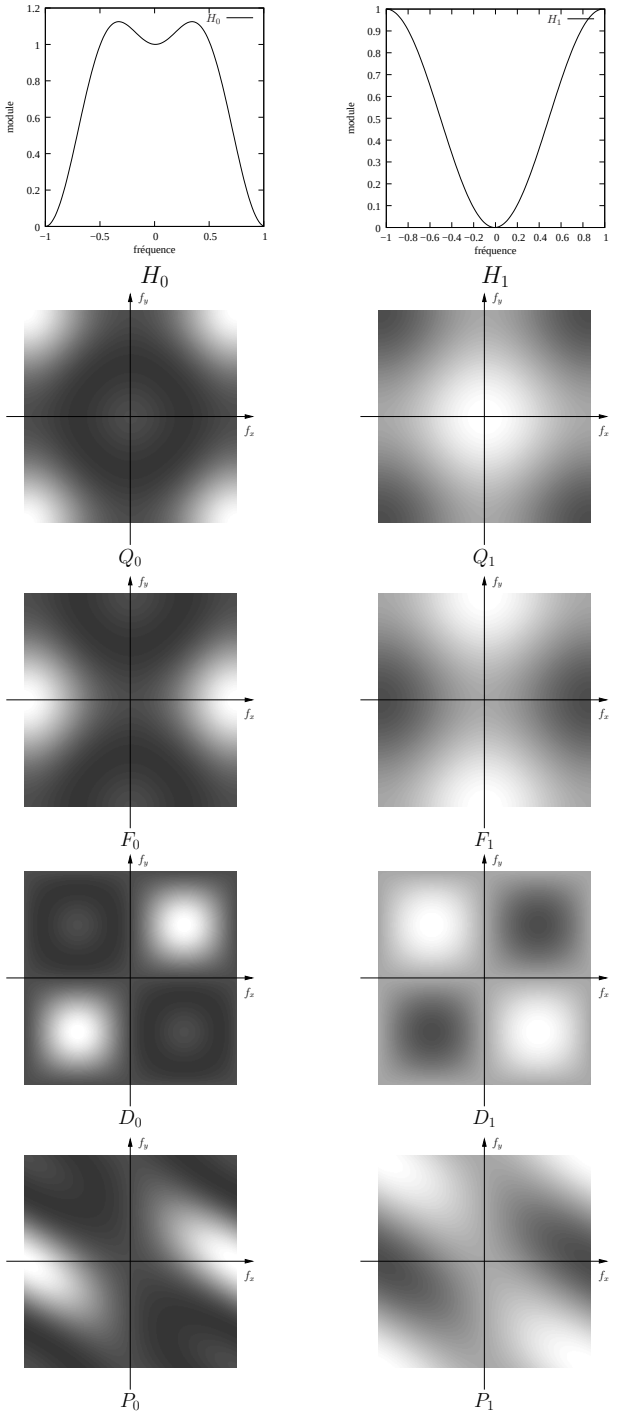

FIG. 3.7: Exemples de paires de filtres quinconce (Q_0, Q_1), éventail (F_0, F_1), damier (D_0, D_1) et parallélogramme (P_0, P_1) issus de la paire de filtres 5/3 (H_0, H_1). La réponse fréquentielle des filtres 2D est représentée par des intensités variant du blanc (module = 0) au noir (module = 2).

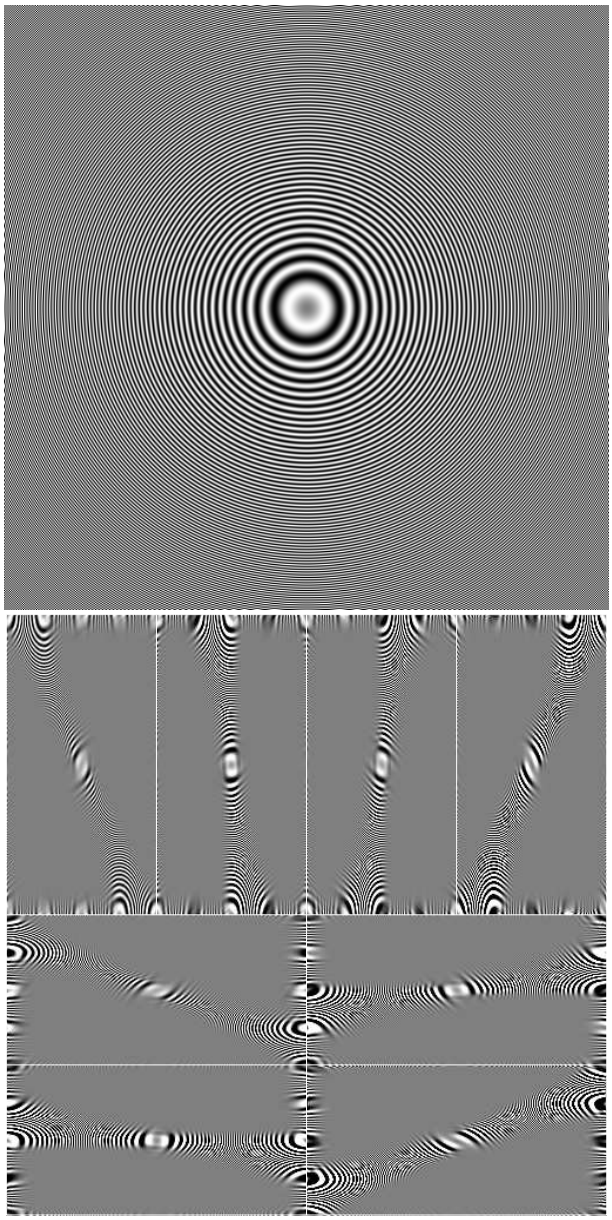

FIG. 3.8: Exemple de décomposition directionnelle en 8 sous-bandes [bas] de l'image zoneplate [haut].

filtre \ niveau	1	2	3	4	5
9/7	0.982948	1.030575	1.051979	1.058014	1.058312
13/11	1.128929	1.243696	1.280835	1.290302	1.291629
MPEG	0.916009	0.931400	0.933205	0.932957	0.931396

TAB. 3.1: Norme des filtres de synthèse de la pyramide laplacienne en fonction du niveau de décomposition.

filtre	0	± 2	± 4	± 6	± 8	± 10
9/7	0.966198	0.055703	-0.051419	0.004094	0	0
13/11	1.274491	-0.017848	-0.078435	0.026715	-0.003437	0.000227
MPEG	0.839085	0.084982	-0.071564	0.029073	-0.004473	0.000447

TAB. 3.2: Orthogonalité intra-échelle des filtres de synthèse de la pyramide laplacienne. Les produits scalaires entre les différentes translatées des filtres sont évalués.

que la reconstruction pseudo-inverse ne règle pas le problème de non orthogonalité de ces filtres qui rend l'optimisation débit-distorsion délicate. Utiliser une pondération des sous-bandes, comme proposée dans [102], permettrait d'effectuer une allocation de débit optimale à haut débit pour ce type de filtres en supposant le bruit de quantification indépendant de la source, mais cette hypothèse est invalide à bas débit.

Pour ces différentes raisons, nous emploierons des filtres 9/7 par la suite dans le cadre de la compression, bien que d'autres filtres pourraient être plus adéquat pour d'autres problèmes de traitement d'image.

3.2.2 Décomposition directionnelle

Pour la décomposition directionnelle, nous avons comparé différentes tailles de filtres à réponse impulsionnelle finie proposés dans [7]. Ces filtres sont obtenus à partir des filtres d'interpolation donnés par les valeurs suivantes :

filtre	0	±1	±3	±5	±7	±9	±11
	±13	±15	±17	±19	±21	±23	±25
	±27	±29	±31				
$H_0^{N=6}$	1.00000	0.62999	0.19296	0.09721	0.05261	0.02724	0.01143
$H_0^{N=16}$	1.00000	0.63376	0.20379	0.11374	0.07281	0.04885	0.03311
	0.02225	0.01463	0.00931	0.00568	0.00329	0.00178	0.00089
	0.00039	0.00014	0.00004				

En appliquant la technique présentée dans [7], les filtres quinconces correspondants sont obtenus. Ces filtres vérifient quasiment la propriété d'orthogonalité comme l'illustre le tableau 3.5. La transformée directionnelle étant composée d'une batterie de filtres possédant tous une composante passe-bas. Les filtres donnés ci-dessus ont la propriété

filtre 9/7	1	2	3	4	5	6
1	1.000000					
2	-0.053428	0.966198				
3	-0.019183	-0.067362	1.062141			
4	-0.002222	-0.023385	-0.067355	1.106900		
5	-0.000377	-0.002680	-0.022927	-0.066580	1.120366	
6	-0.000057	-0.000451	-0.002610	-0.022502	-0.066301	1.123926

filtre 13/11	1	2	3	4	5	6
1	1.000000					
2	-0.048333	1.274491				
3	-0.010756	-0.053974	1.546836			
4	-0.000844	-0.011017	-0.049373	1.640777		
5	-0.000096	-0.000856	-0.009528	-0.047634	1.665848	
6	-0.000009	-0.000095	-0.000747	-0.009004	-0.047165	1.672206

filtre MPEG	1	2	3	4	5	6
1	1.000000					
2	0.182592	0.839085				
3	-0.004953	0.197327	0.867563			
4	0.004893	-0.003860	0.199507	0.871112		
5	0.001213	0.004978	-0.004057	0.199670	0.871383	
6	0.000164	0.001204	0.004954	-0.004081	0.199681	0.871401

TAB. 3.3: Orthogonalité inter-échelle des filtres de synthèse de la pyramide laplacienne. Le produit scalaire entre la fonction de base au centre de la sous-bande basse et les différentes fonctions de bases d'échelle au centre des sous-bandes hautes est évalué.

#orientations \ sous-bande	0	1	2	3	4	5	6	7
	8	9	10	11	12	13	14	15
N = 6								
2^1	1.3390	0.7683						
2^2	1.7887	1.0317	1.0317	0.5880				
2^3	2.3502	1.3886	1.4731	0.7552	1.5210	0.7180	0.7605	0.4664
2^4	2.9901	1.8596	2.0095	0.9934	2.1537	1.0402	1.0716	0.5650
	2.3100	1.0157	1.0439	0.5120	1.1087	0.5395	0.5803	0.3832
N = 16								
2^1	1.3667	0.7464						
2^2	1.8660	1.0214	1.0214	0.5561				
2^3	2.5195	1.4024	1.4598	0.7361	1.4936	0.7101	0.7392	0.4268
2^4	3.3119	1.9274	2.0229	0.9935	2.1222	1.0244	1.0540	0.5355
	2.2502	0.9974	1.0260	0.5020	1.0733	0.5197	0.5451	0.3393

TAB. 3.4: Norme des filtres de synthèse du banc de filtres directionnels.

de conserver la moyenne du signal traité, mais ne sont en revanche aucunement normalisés. Dans le cadre de la compression, cette propriété est essentielle pour effectuer correctement l'allocation de débit entre les différentes sous-bandes directionnelles d'une transformée en contourlettes. Les sous-bandes sont donc pondérées par la norme des filtres, figurant dans le tableau 3.4, avant quantification.

3.3 Codage par ondelettes séparables et contourlettes

Bien que les contourlettes soient très efficaces pour l'approximation des contours de l'image en un faible nombre de coefficients, leur performance sur les images naturelles par rapport aux ondelettes séparables a tendance à décroître lorsque le nombre de coefficients augmente. En effet, dans le cas extrême du codage sans pertes, l'énergie de l'image est dispersée dans un plus grand nombre de coefficients et le codage par ondelettes donne de meilleurs résultats. Comme les images naturelles ont généralement plus d'énergie dans les basses fréquences, pour un débit donné il y a souvent bien plus de coefficients conservés dans les échelles associées que dans les échelles fines correspondant aux hautes fréquences. Ceci suggère d'utiliser une décomposition en ondelettes pour les échelles grossières et une décomposition en contourlettes pour les échelles fines. Comme les sous-bandes de basse fréquence de la transformée en contourlettes sont exactement identiques aux sous-bandes de basse fréquence de la transformée en ondelettes séparables, chaque niveau peut être décomposé indépendamment soit en 3 sous-bandes d'ondelettes, soit en 2^n sous-bandes directionnelles. De plus, comme l'énergie a tendance à décroître des échelles grossières aux échelles fines, il suffit de coder l'index du niveau à partir duquel la transition entre contourlette et ondelette s'effectue. Le coût de codage de cet index est alors négligeable.

N=6	0	1	2	3	4	5	6
0	1.000000						
1	-0.022105	1.000000					
2	0.004697	-0.016583	1.000000				
3	-0.006652	-0.009149	-0.002874	1.000000			
4	-0.007921	0.009781	0.001065	-0.000062	1.000000		
5	-0.011792	0.000377	0.001177	-0.002588	0.008779	1.000000	
6	-0.002510	0.001379	-0.001353	-0.008813	0.022640	0.034303	1.000000
7	0.000034	0.002800	0.004083	-0.005658	0.007528	0.011087	0.035621

N=16	0	1	2	3	4	5	6
0	1.000000						
1	-0.009244	1.000000					
2	0.004310	-0.006448	1.000000				
3	-0.001800	-0.003544	-0.002270	1.000000			
4	-0.003761	0.003867	0.000714	-0.000014	1.000000		
5	-0.004427	0.000142	0.000315	-0.001213	0.005018	1.000000	
6	-0.001005	0.000137	-0.000245	-0.004344	0.011759	0.015691	1.000000
7	0.000017	0.001145	0.003294	-0.002947	0.004138	0.002080	0.017026

TAB. 3.5: Produit scalaire entre les filtres des sous-bandes directionnelles normalisés.

Le niveau à partir duquel la transition entre contourlettes et ondelettes s'effectue peut être fixé arbitrairement ou donné par un critère débit-distorsion. Dans ce dernier cas, la configuration optimale est déterminée pour un débit cible donné en minimisant le lagrangien $J = D+\lambda R$, où D est la distorsion et R le débit dépendant de la configuration choisie. Pour des raisons de complexité, cette minimisation est effectuée dans le domaine transformé et R est estimé à partir de l'entropie absolue des coefficients d'ondelettes et de contourlettes de chaque sous-bande. Notons que, pour une décomposition sur L niveaux, il serait également possible d'obtenir R en codant les sous-bandes dans les $L+1$ configurations possibles à l'aide d'un codeur de sous-bande progressif (comme EZBC ou EBCOT) puis en tronquant les flux obtenus a posteriori pour satisfaire la contrainte de débit. Le coût induit en complexité nous paraît cependant prohibitif par rapport à une simple estimation par entropie absolue.

La transformée en contourlettes forme une frame étroite dans le cas où des filtres orthogonaux sont utilisés à la fois pour la décomposition pyramidale et la décomposition directionnelle [4]. La transformée en ondelettes formant une base orthogonale dans ce même cas, l'ensemble des deux transformées forme une frame étroite également lorsque tous les filtres utilisés sont orthogonaux. Nous avons vu dans la partie précédente que les filtres d'analyse directionnelle sont quasiment orthogonaux, de même que les filtres 9/7 utilisés à la fois pour la décomposition pyramidale et la décomposition en ondelettes. La reconstruction est obtenue en utilisant l'opérateur de synthèse pseudo inverse approximé par l'opérateur transposé de l'opérateur d'analyse. Parmi tous les opérateurs

de reconstruction possibles, cet opérateur minimise l'erreur quadratique de reconstruction en présence de bruit blanc additif dans le domaine transformé. Ainsi, une fois les filtres normalisés, nous pouvons supposer que la distorsion introduite dans le domaine transformé se retrouve directement dans l'image. Les normes des filtres de synthèse en contourlettes, combinant les filtres [7] et 9/7, sont données dans le tableau 3.6 pour plusieurs niveaux de décomposition pyramidale et d'analyse directionnelle. La norme du filtre passe-bas correspondante se trouve dans le tableau 3.1.

La transformée en contourlettes étant surtout efficace à bas débit, l'opération de quantification est similaire à un seuillage, ce qui permet d'approximer l'opposé de la pente débit-distorsion par $\lambda \approxeq \frac{3\Delta^2}{4\gamma_0}$ [84], où Δ est le pas du quantificateur uniforme par zone morte utilisé et γ_0 est une constante représentant le coût de codage d'un coefficient non nul, fixée à $\gamma_0 = 7$. Le pas de quantification est alors le même pour toutes les sous-bandes et l'optimisation revient simplement à chercher le pas Δ et le niveau de transition correspondant qui mène au débit R désiré. Le niveau de transition approprié pour un pas Δ donné est déterminé par une recherche exhaustive des configurations minimisant la distortion parmi les $L+1$ choix possibles, où L est le nombre de niveaux de décomposition.

Cette nouvelle transformée fournit une analyse directionnelle dans les hautes fréquences tout en améliorant les performances en compression à bas débit par rapport à l'ondelette. De plus, un décodage progressif est toujours possible. La figure 3.10 illustre la partition fréquentielle effectuée par cette transformée en supposant des filtres idéaux. Notons que, bien qu'à notre connaissance aucune publication n'y fasse référence, une implémentation de la transformée combinant les contourlettes et les ondelettes, indépendante de la nôtre[2] a été fournie par les auteurs des contourlettes postérieurement à ce travail[3].

3.4 Optimisation de la transformée en contourlettes

Nous avons vu dans la section précédente que la transformée en contourlettes et la transformée hybride fournissent une frame d'analyse étroite pour des filtres orthogonaux. L'opérateur de synthèse pseudo-inverse donne alors la reconstruction linéaire optimale en présence de bruit additif indépendant. Dans une application de compression, la quantification, exprimée au travers de la fonction non-linéaire $Q^{-1} \circ Q$ appliquée à chaque élément de $\mathsf{y} = \mathsf{Ax}$, introduit un bruit de distorsion d, tel que $\hat{\mathsf{y}} = \mathsf{Ax} + \mathsf{d}$, qui n'est ni additif ni indépendant. Comme la transformée est redondante, il existe plusieurs vecteurs y permettant de reconstruire le même signal x. Soit $\mathcal{S} = \{\mathsf{y} = \mathsf{Ax}, \mathsf{x} \in \mathbb{R}^N\}$ l'espace vectoriel image de l'opérateur d'analyse A et $\mathcal{S}^\perp$ son complément orthogonal dans $\mathbb{R}^M$. Alors,

$$\forall \mathsf{y} \in \mathbb{R}^M, \exists ! (\mathsf{y}^{\mathcal{S}}, \mathsf{y}^{\perp}) \in \mathcal{S} \times \mathcal{S}^{\perp}, \mathsf{y} = \mathsf{y}^{\mathcal{S}} + \mathsf{y}^{\perp},$$

avec $\mathsf{y}^{\mathcal{S}} = \mathsf{ABy} = \mathsf{P}^{\mathcal{S}}\mathsf{y}$ et $\mathsf{y}^{\perp} = (\mathsf{I} - \mathsf{AB})\mathsf{y} = \mathsf{P}^{\perp}\mathsf{y}$. Puisque la transformée est redondante, la transformée de l'erreur $\mathsf{A}(\mathsf{x} - \hat{\mathsf{x}}) = \mathsf{A}(\mathsf{x} - \mathsf{B}\hat{\mathsf{y}}) = \mathsf{A}(\mathsf{x} - \mathsf{B}(\mathsf{y} + \mathsf{d})) = \mathsf{A}(\mathsf{x} - \mathsf{BAx} - \mathsf{Bd}) =$

[2] http ://www.irisa.fr/temics/Equipe/Chappelier/contourlets.tar.gz
[3] http ://www.ifp.uiuc.edu/%7Eminhdo/software/contourlet_toolbox.tar

sous-bande	0	1	2	3	4	5	6	7
niveau 1	8	9	10	11	12	13	14	15
2^0	0.7598							
2^1	1.0681	0.7101						
2^2	1.5576	0.9223	0.8856	0.5139				
2^3	2.0668	1.2000	1.3126	0.6790	1.3143	0.6110	0.6671	0.4067
2^4	2.5917	1.6505	1.7263	0.8662	1.9330	0.9198	0.9470	0.5192
	2.0041	0.8715	0.8802	0.4411	0.9800	0.4692	0.4956	0.3390

sous-bande	0	1	2	3	4	5	6	7
niveau 2	8	9	10	11	12	13	14	15
2^0	0.7538							
2^1	1.0680	0.7302						
2^2	1.5913	0.9298	0.9087	0.5190				
2^3	2.1423	1.2105	1.3243	0.6810	1.3699	0.6139	0.6702	0.4101
2^4	2.6464	1.7244	1.7796	0.8506	1.9438	0.9293	0.9756	0.5041
	2.0891	0.9072	0.9038	0.4312	0.9816	0.4718	0.5072	0.3364

sous-bande	0	1	2	3	4	5	6	7
niveau 3	8	9	10	11	12	13	14	15
2^0	0.7759							
2^1	1.0982	0.7478						
2^2	1.6313	0.9534	0.9323	0.5327				
2^3	2.2043	1.2370	1.3512	0.7021	1.4090	0.6278	0.6843	0.4228
2^4	2.7036	1.7559	1.8059	0.8654	1.9662	0.9470	0.9957	0.5212
	2.1343	0.9249	0.9176	0.4390	0.9932	0.4811	0.5173	0.3485

sous-bande	0	1	2	3	4	5	6	7
niveau 4	8	9	10	11	12	13	14	15
2^0	0.7826							
2^1	1.1073	0.7531						
2^2	1.6432	0.9599	0.9393	0.5367				
2^3	2.2034	1.2342	1.3494	0.7047	1.4116	0.6287	0.6833	0.4274
2^4	2.8462	2.1272	1.8988	1.2989	2.0806	1.2993	1.0859	0.8085
	2.1805	1.1696	0.9830	0.5943	1.0349	0.5634	0.5332	0.3932

TAB. 3.6: Norme des filtres de synthèse de la transformée en contourlettes.

$-\mathsf{ABd} = -\mathsf{P}^{\mathcal{S}}\mathsf{d} = -\mathsf{d}^{\mathcal{S}}$ diffère de la distorsion d initialement introduite par la non-linéarité $Q^{-1} \circ Q$. En approximant la quantification à zone morte par un seuillage, hypothèse uniquement valide à bas débit dans la mesure ou la transformée inverse est linéaire et proche de l'orthonormalité, $\mathsf{d}^{\mathcal{S}}$ peut aussi contenir des coefficients non nuls à des positions où $\hat{y}$ est non nul. Ainsi, soustraire $\mathsf{d}^{\mathcal{S}}$ à $\hat{\mathsf{y}}$ tend à augmenter l'énergie des coefficients conservés tout en réduisant l'énergie des coefficients annulés par le seuillage. De plus, $\mathsf{B}(\hat{\mathsf{y}}-\mathsf{d}^{\mathcal{S}}) = \mathsf{B}(\mathsf{y}+\mathsf{d}^{\perp})$ est toujours égal à x. Pour le même nombre de coefficients non nuls, le vecteur creux obtenu par seuillage de $\hat{y} - \mathsf{d}^{\mathcal{S}}$ est plus proche de y que $\hat{\mathsf{y}}$, réduisant par là même l'erreur de reconstruction.

L'algorithme de projection itératif conçu dans [56] pour la transformée redondante en ondelettes complexes est basé sur cette observation, et peut également s'appliquer dans le contexte des contourlettes ou de l'approche hybride proposée. L'image x est transformée en y_0. Chaque boucle consiste en les étapes suivantes : y_i est quantifié en $\hat{\mathsf{y}}_i$, ce qui introduit une distorsion d_i. Le signal $\hat{\mathsf{x}}_i = \mathsf{B}\hat{\mathsf{y}}_i$ est reconstruit par transformée inverse et soustrait de x pour obtenir l'image d'erreur e_i. Les coefficients de mise à jour u_i sont obtenus par transformée de e_i et ajoutés à $\hat{\mathsf{y}}_i$ pour former y_{i+1}. Ce processus est itéré un certain nombre de fois jusqu'à obtenir la convergence. En remarquant que $\mathsf{y}_i^{\mathcal{S}} = \mathsf{y}_0$ on obtient la relation récursive suivante entre y_i et y_{i+1} :

$$\begin{aligned}\mathsf{u}_i &= \mathsf{A}(\mathsf{x} - \mathsf{B}\hat{\mathsf{y}}_i) = \mathsf{y}_0 - \mathsf{y}_i^{\mathcal{S}} - \mathsf{d}_i^{\mathcal{S}} = -\mathsf{d}_i^{\mathcal{S}},\\ \mathsf{y}_{i+1} &= \hat{\mathsf{y}}_i + \mathsf{u}_i = \mathsf{y}_i + \mathsf{d}_i - \mathsf{d}_i^{\mathcal{S}} = \mathsf{y}_i + \mathsf{d}_i^{\perp}.\end{aligned}$$

3.4.1 Etude de la convergence

La convergence de l'algorithme proposé peut s'étudier en utilisant la théorie des projections sur espaces convexes [103] (POCS). Une partie $\mathcal{E}$ d'un espace vectoriel est dite convexe si pour tout couple de vecteurs de $\mathcal{E}$, les vecteurs du segment formé par ce couple sont également dans $\mathcal{E}$. Dans le cas étudié, nous nous intéressons en particulier à l'ensemble $\mathcal{S}$ sur lequel le vecteur y est projeté par la transformée et l'ensemble $\mathcal{Q}$ sur lequel y est projeté par la quantification. La théorie des POCS nous donne deux résultats dans le cas ou $\mathcal{S}$ et $\mathcal{Q}$ sont convexes. Soit les ensembles sont d'intersection non vide, auquel cas les projections itératives sur chacun de ces ensembles à tour de rôle convergent vers un point fixe $\mathsf{y}^{\star} \in \mathcal{S} \cap \mathcal{Q}$. Soit les ensembles sont disjoints, et l'algorithme entre dans un cycle limite alternant entre le point de $\mathcal{S}$ le plus proche de $\mathcal{Q}$ et le point de $\mathcal{Q}$ le plus proche de $\mathcal{S}$ (Fig. 3.11).

L'ensemble $\mathcal{S}$ est trivialement convexe car il forme un espace vectoriel. Le cas de $\mathcal{Q}$ est plus délicat car $\mathcal{Q}$ est formé par un nuage de vecteurs correspondant aux centroïdes de quantification, qui n'est pas du tout convexe (Fig. 3.12). Il est cependant possible d'approximer l'opération de quantification par un seuillage des coefficients, ce qui est plus ou moins valide pour un quantificateur uniforme à zone morte à bas débit couplé à une transformée inverse quasi-orthonormée. Dans ce cas, on considère l'ensemble $\mathcal{T}_{\Delta}$ des coefficients d'amplitude supérieure à un seuil Δ, correspondant au pas de quantification, après la transformée initiale. En supposant que cet ensemble est fixe au cours des

itérations, c'est à dire que les coefficients d'amplitude inférieure au seuil reste quantifiés à zéro tandis que ceux d'amplitude supérieure au seuil voient leur amplitude croître, cet ensemble est alors convexe. En effet si deux coefficients sont supérieurs au seuil, les valeurs intermédiaires le sont également. Le pas de quantification étant choisi pour que tous les coefficients ne soient pas nul, les deux ensembles $\mathcal{S}$ et $\mathcal{T}_\Delta$ s'intersectent et l'algorithme converge donc vers un point fixe.

La convergence n'étant assurée théoriquement que si l'ensemble des coefficients conservés est identique à chaque itération et que la non-linéarité est un seuillage, il est cependant possible d'appliquer l'algorithme dans le cas général. En effet, même si l'algorithme entre dans un cycle, celui-ci correspond à deux solutions proches dans l'espace $\mathcal{S}$ ou l'ensemble $\mathcal{Q}$ dans la mesure ou le pas de quantification est suffisamment fin. De plus, le nombre de coefficients quantifiés à zéro ne varie que légèrement au cours des itérations de sorte que l'ensemble $\mathcal{Q}$ est relativement invariant. Nous supposerons donc par la suite que l'algorithme converge vers un point fixe également dans les conditions réelles de quantification, hypothèse validée par les résultats expérimentaux. Dans ce cas, $d_i^\perp$ tend vers zero, ce qui signifie que le bruit est entièrement contenu dans l'espace $\mathcal{S}$.

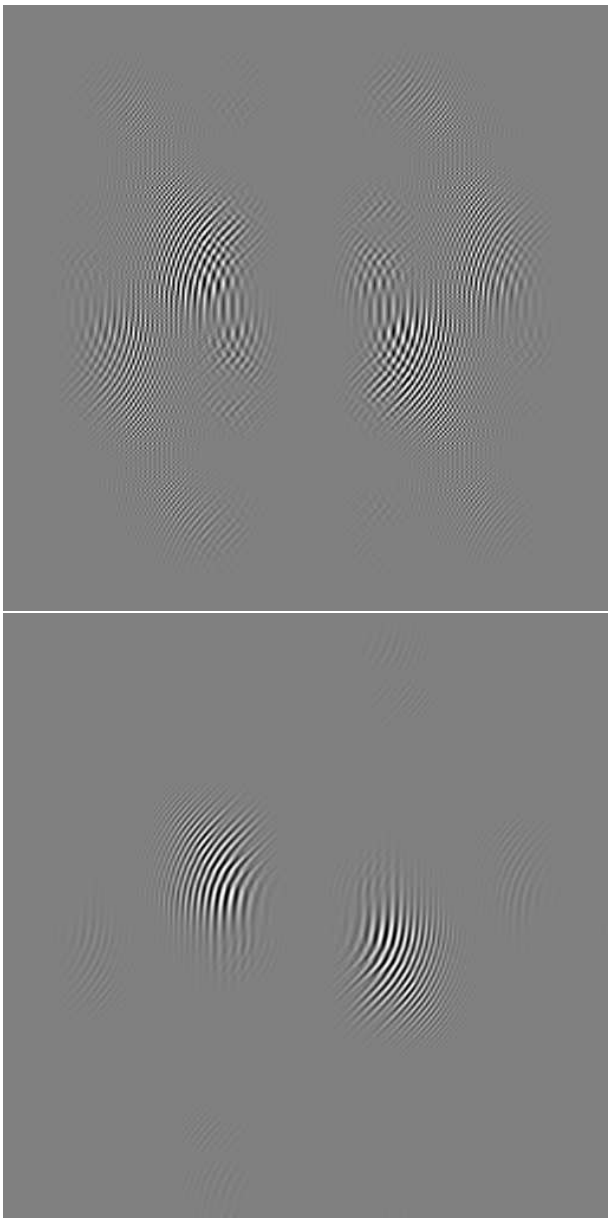

FIG. 3.9: [de haut en bas] Impact des différents filtres de la pyramide laplacienne sur la sélectivité des sous-bandes pour les filtres 9/7 et MPEG. La sous-bande 3/4 du second niveau de décomposition en contourlettes de l'image *zoneplate* est représentée. Le filtre MPEG offre la meilleure sélectivité tout en étant loin de l'orthogonalité.

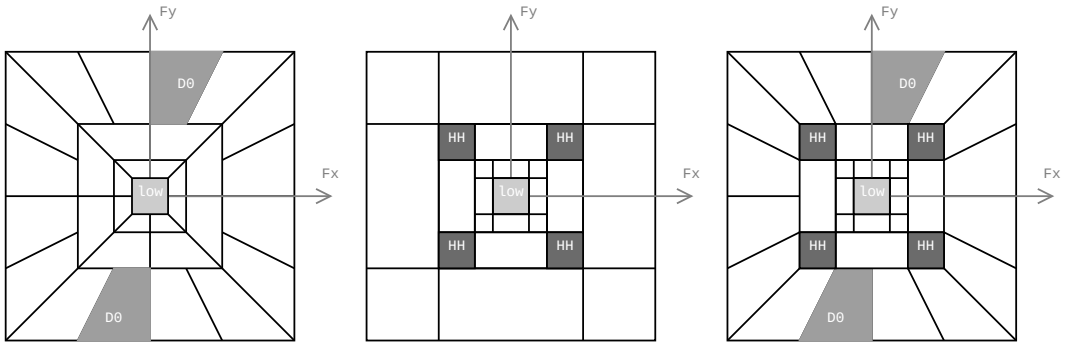

FIG. 3.10: Partition fréquentielle de la contourlette, l'ondelette séparable et du schéma hybride. L'une des sous-bandes directionnelles du 1er niveau ainsi que la sous-bande HH du second niveau sont sur-lignées en gris.

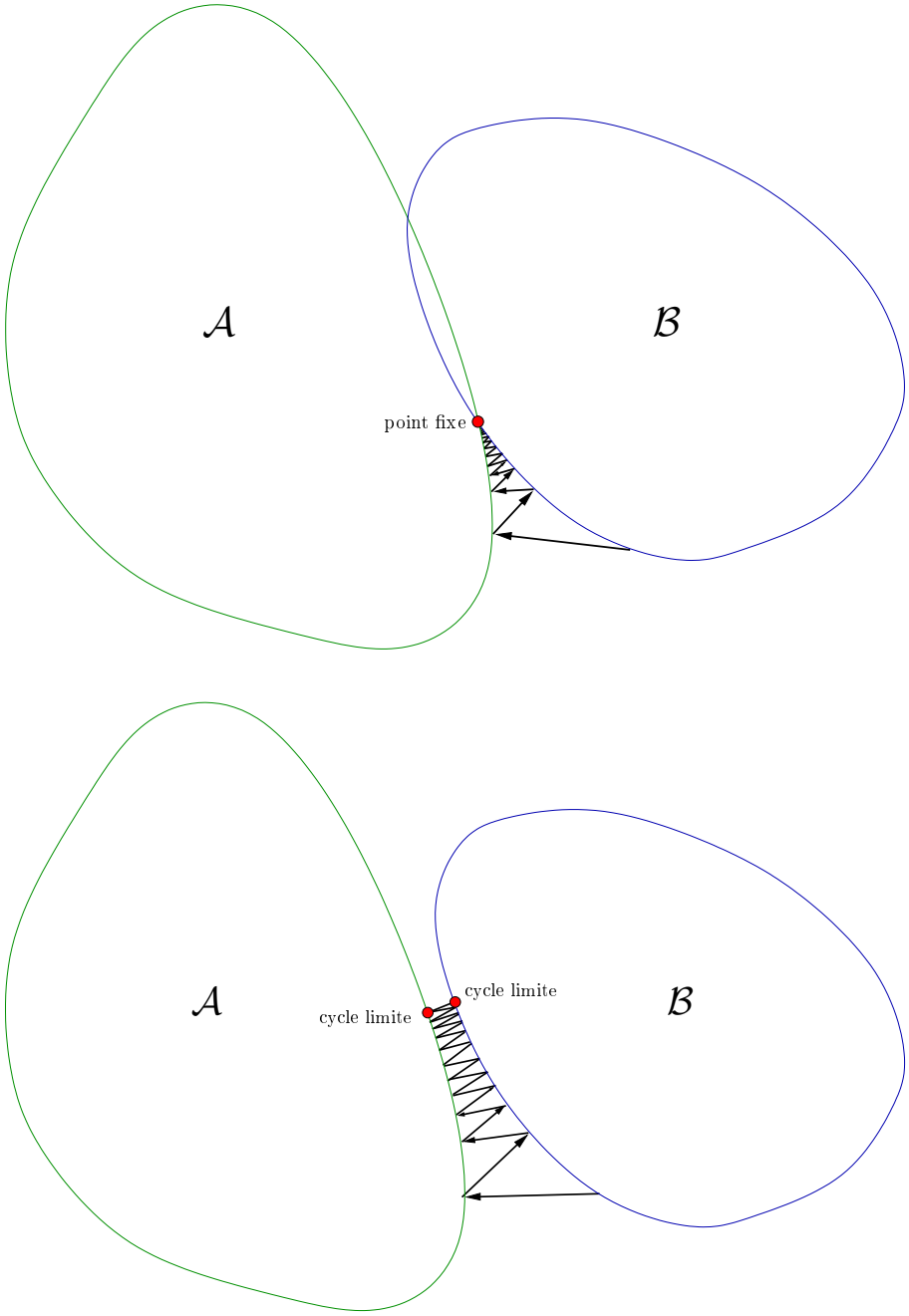

FIG. 3.11: Projections sur espaces convexes. [haut] Cas d'ensembles d'intersection non vide. L'algorithme converge vers un point fixe. [bas] Cas d'ensembles disjoints. L'algorithme entre dans un cycle limite.

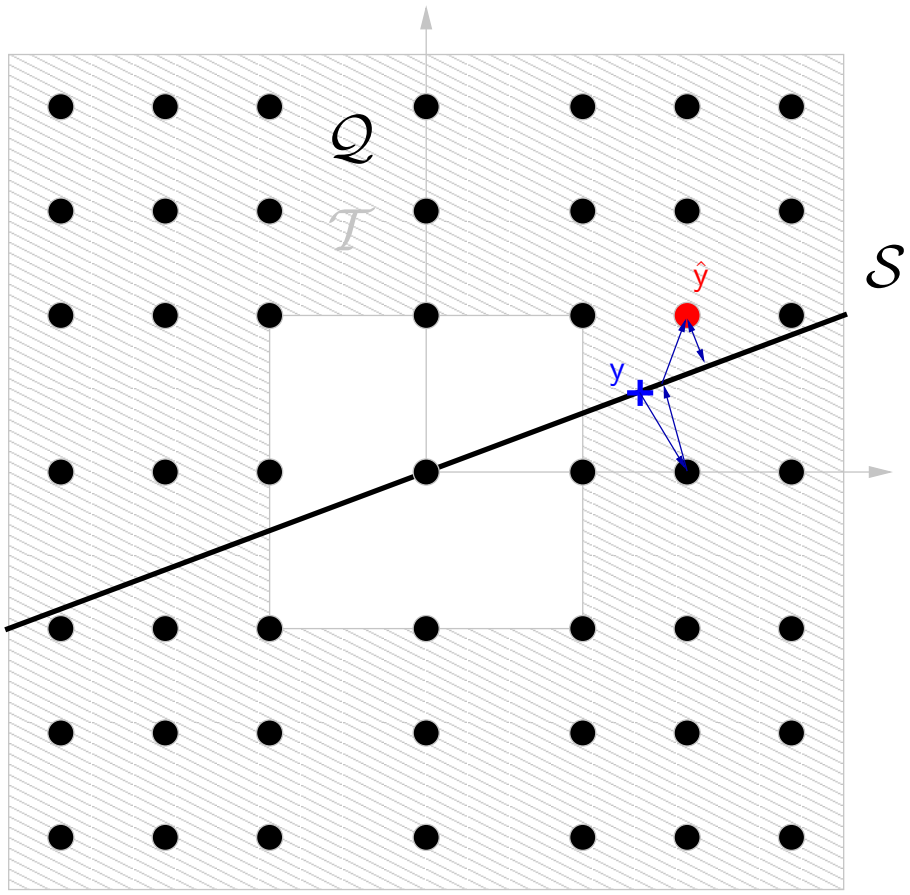

FIG. 3.12: Illustration du principe de projections sur espaces convexes pour la quantification des contourlettes. L'ensemble $\mathcal{S}$ est représenté par une droite (de dimension inférieure au plan) sur laquelle se trouve le vecteur $\mathbf{y}$ initial. Après plusieurs projections, l'algorithme converge et le centroïde retenu n'est pas forcément le plus proche de $\mathbf{y}$ dans l'absolu.

3.5 Application à la compression d'images

Dans le cadre de l'application de la transformée en ondelettes et contourlettes à la compression d'image, nous avons utilisé les filtres 9/7 biorthogonaux [12] pour la décomposition pyramidale et la décomposition par ondelettes. Nous avons tout d'abord étudié la performance en compression de la transformée hybride par rapport à la transformée en ondelettes séparable en utilisant une quantification par zone morte de même pas dans chaque sous-bande. La distribution des coefficients quantifiés est supposée stationnaire au sein d'une sous-bande et est estimée par l'histogramme, le débit étant alors obtenu par l'entropie absolue des coefficients quantifiés. Nous avons également appliqué le principe de projection sur ensembles convexes à ce schéma simple pour en évaluer le gain de codage. Dans ces expériences, les nombre de décompositions multi-échelles est fixé à 4 niveaux. Les filtres utilisés pour la décomposition directionnelle sont les filtres à réponse impulsionnelle finie de Phoong [7], avec $N = 16$. Le nombre de directions d'analyse dans chaque sous-bande a été fixé à 16. Comme la bande basse est identique dans toutes les transformées considérées, elle n'a pas été quantifiée ni prise en compte dans le calcul de l'estimation du débit.

Le tableau 3.7 montre le gain en performance de compression par rapport au codage par transformée en ondelettes séparables et par transformée en contourlettes pour quelques images. Ces images contiennent des caractéristiques directionnelles dans les hautes fréquences qui sont représentées efficacement par le banc de filtres directionnels aux échelles fines. Les fréquences les plus basses sont codées quant à elles efficacement au moyen d'ondelettes séparables. Ceci résulte en un gain de 0.2 à 0.5 dB par rapport à la transformée en contourlettes classique. La procédure d'optimisation itérative permet d'obtenir un gain additionnel de 0.2 dB (Fig. 3.13). Pour les images ne contenant pas de caractéristiques directionnelles fortes, la transformée hybride devient alors équivalente à une transformée en ondelettes séparables, ce qui garantit un niveau de performance au moins égal aux performances de la transformée en ondelettes séparable dans le cas d'un codage non progressif.

La figure 3.16 illustre le gain apporté par la transformée hybride proposée pour différentes valeurs d'entropie. Un gain de 0.5 à 1 dB par rapport à la transformée en ondelettes séparables est observé à bas débit. Notons que la plupart du gain entre la version itérée et non-itérée de la transformée hybride s'obtient après la première itération. Ces itérations étant coûteuse du fait de l'application de la transformée et de la transformée inverse, ces résultats montrent qu'il est inutile d'itérer plus d'une fois la projection sur ensembles convexes.

Nous avons également effectué une série de tests de compression à l'aide du codeur de sous-bandes EZBC. Ce codeur a été adapté à la transformée en contourlettes en conservant les contextes de codage des sous-bandes d'ondelettes. Les sous-bandes directionnelles correspondant aux orientations $[-\frac{\pi}{4}, \frac{\pi}{4}]$ ont été codées en tant que sous-bandes horizontales, tandis que les sous-bandes correspondant aux orientations $[\frac{\pi}{4}, \frac{3\pi}{4}]$ ont été codées en tant que sous-bandes verticales. Le contexte inter-échelle a été supprimé pour simplifier l'adaptation, son impact sur le débit total étant négligeable. Nous avons utilisé une décomposition sur 5 niveaux. Le premier niveau a été décomposé en 16 sous-bandes

image	bpp	ondelette	contourlette	hybride	hybride itérée
barbara	0.200	26.62	26.83	27.34	27.57
bike	0.224	23.31	23.14	23.31	23.87
zoneplate*	0.232	17.31	17.43	18.08	18.21

TAB. 3.7: Valeurs de PSNR (dB) obtenues avec les différentes transformées appliquées sur deux images naturelles 512x512 et une image synthétique (*) constituée de cercles concentriques. Le débit est estimé par l'entropie absolue des sous-bandes quantifiées.

directionnelles, tandis que le second niveau a été décomposé en 8 sous-bandes directionnelles. Les autres niveaux, de plus basse fréquences, ont été décomposés en ondelettes séparables. Notons que dans le cadre d'un codage progressif, cette configuration est fixée pour toute la plage de débits considérée. Nous avons réalisé une optimisation débit-distorsion a posteriori, en tronquant les flux binaires issus du codage de chaque sous-bande de manière optimale par rapport au débit cible. Cette optimisation est réalisée de manière similaire à celle effectuée dans JPEG2000 par le codeur EBCOT. Les figures 3.14 et 3.15 montrent les performances débit-distorsion du schéma hybride décrit par rapport aux performances d'une transformée en ondelette séparables. Un léger gain est observé à bas débit pour les images possédant des caractéristiques directionnelles. Ces performances chutent toutefois à haut débit, du fait de la redondance de la transformée en contourlettes. Notons cependant qu'une adaptation plus fine des contextes de codage de EZBC à la transformée en contourlettes pourrait permettre d'obtenir un gain plus important à bas débit. Les figures 3.17 et 3.18 montrent les images *zoneplate* et *barbara* reconstruites après décodage pour les deux transformées considérées. Bien que le gain objectif en termes de PSNR ne soit pas flagrant, la qualité visuelle des images reconstruite est meilleure car l'effet de pixelisation et de rebonds à proximité des contours est réduit par la transformée hybride et les contours sont mieux préservés.

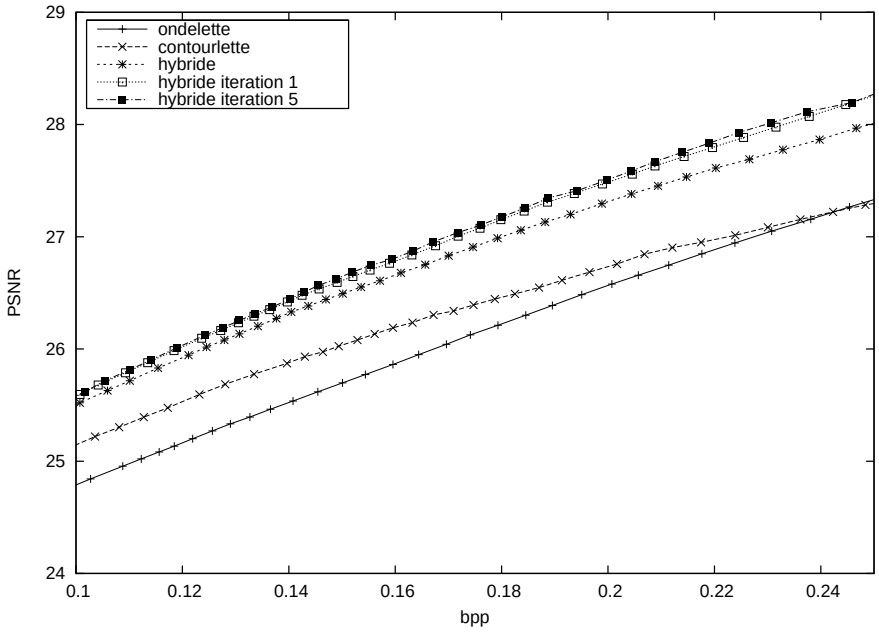

FIG. 3.13: Performance débit-distorsion des transformées en ondelettes séparable, en contourlette, hybride, et hybride itérée sur l'image *barbara*.

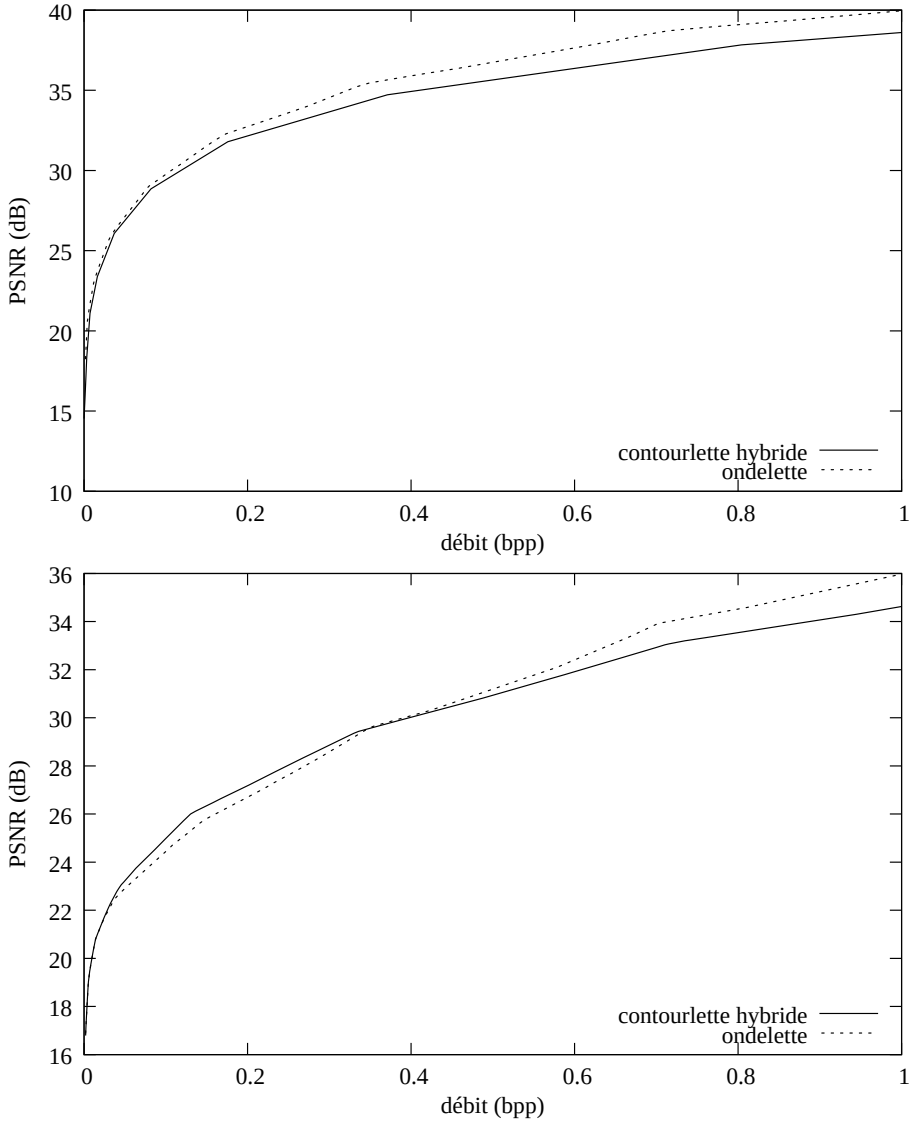

FIG. 3.14: Performance débit-distorsion de la transformée hybride et de la transformée en ondelettes sur 5 niveaux pour les images *lena* [haut] et *barbara* [bas]. La transformée hybride utilise une analyse sur 16 et 8 directions respectivement pour le premier et le second niveau. Les niveaux suivants sont décomposés en ondelettes séparables.

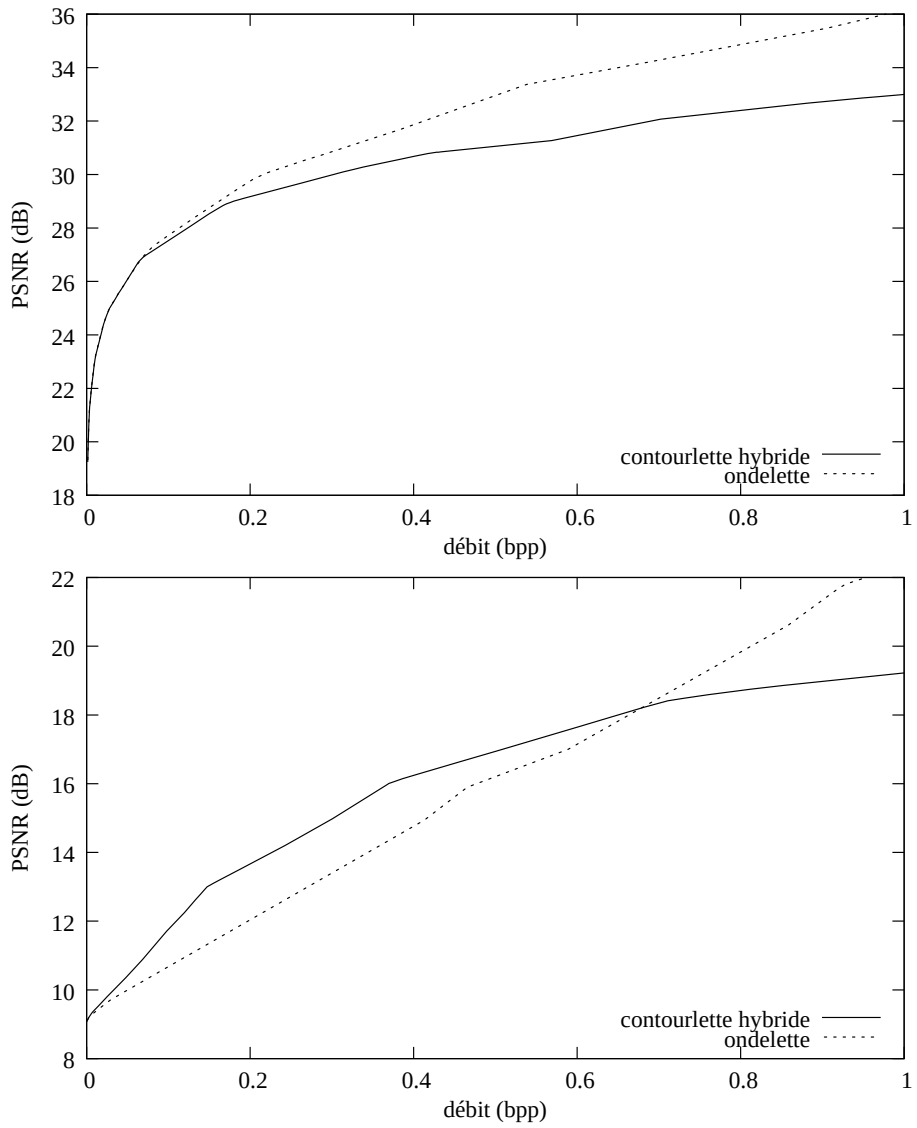

FIG. 3.15: Performance débit-distorsion de la transformée hybride et de la transformée en ondelettes sur 5 niveaux pour les images *goldhill* [haut] et *zoneplate* [bas]. La transformée hybride utilise une analyse sur 16 et 8 directions respectivement pour le premier et le second niveau. Les niveaux suivants sont décomposés en ondelettes séparables.

FIG. 3.16: [haut] Transformée en ondelettes séparable 9/7 sur 4 niveaux de l'image *barbara*. Pour une entropie des coefficients quantifiés des sous-bandes de hautes fréquences égale à 0.2 bpp, le PSNR vaut 26.62 dB. [bas] Transformée hybride sur 4 niveaux de l'image *barbara*. Pour une entropie des coefficients quantifiés des sous-bandes de hautes fréquences égale à 0.2bpp, le PSNR vaut 27.57 dB. La sous-bande de basse fréquence est identique pour les deux transformées et codée parfaitement.

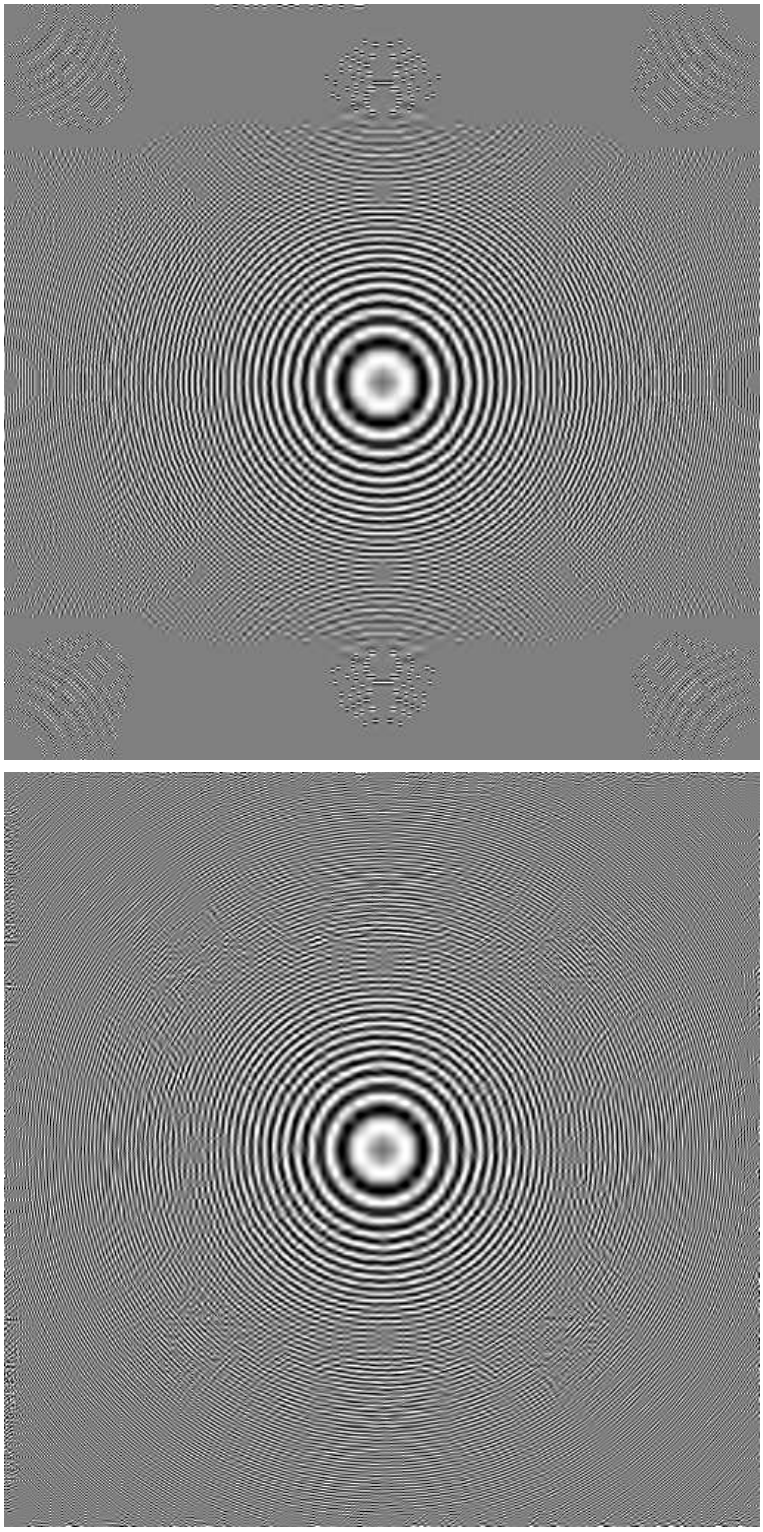

FIG. 3.17: [haut] Codage par EZBC d'une transformée en ondelettes séparable 9/7 sur 5 niveaux de l'image *zoneplate* à 0.25 bpp, PSNR = 11.64 dB. [bas] Codage par transformée hybride et EZBC sur 5 niveaux de l'image *zoneplate* à 0.20bpp, PSNR = 13.86 dB.

FIG. 3.18: [haut] Codage par EZBC d'une transformée en ondelettes séparable 9/7 sur 5 niveaux de l'image *barbara* à 0.20 bpp, PSNR = 26.59 dB. [bas] Codage par transformée hybride et EZBC sur 5 niveaux de l'image *barbara* à 0.20bpp, PSNR = 26.99 dB.

3.6 Application à la compression vidéo

Nous avons étudié l'adaptation de la transformée en contourlettes au schéma de codage vidéo par ondelettes compensées en mouvement nommé *Motion Compensated Wavelet Coder* (MCWC) et basé en partie sur la norme H.264 (Fig. 3.19). Il s'agit d'un codeur vidéo dans lequel la séquence originale (typiquement CIF[4] 30Hz) est codée de manière progressive à la fois en résolution spatiale[5] et temporelle. La séquence originale est filtrée temporellement à l'aide d'ondelettes compensées en mouvement (MCTF) puis décimée, en ne conservant que la sous bande de basse fréquence spatiale et temporelle, en une séquence de basse résolution (typiquement QCIF[6] 7.5Hz) appelée *couche de base*. Cette *couche de base* est codée à l'aide du codeur H.264[7] à débit constant. Différentes *couches d'améliorations* sont ensuite construites en s'appuyant sur l'information contenue dans la couche de base pour ne coder que les détails supplémentaires permettant d'augmenter la qualité, la résolution spatiale et la résolution temporelle de la séquence. Nous ne nous intéressons pas ici aux problèmes de compensation de mouvement et de progressivité temporelle, pour nous concentrer sur la progressivité spatiale.

Les filtres de décimation et d'interpolation utilisés pour le changement de résolution spatiale formant une pyramide laplacienne, il nous est paru intéressant d'étudier la possibilité d'utiliser des contourlettes au lieu des ondelettes séparables pour représenter les détails spatiaux de la sous-bande de basse fréquence temporelle. En effet, cette sous-bande contient principalement des informations de contours qu'il parait plus naturel de coder à l'aide d'une transformée directionnelle. Par rapport au schéma de codage actuel, cette expérience requiert trois modifications du codeur.

Tout d'abord, la reconstruction est effectuée par simple ajout de la bande basse sur-échantillonnée et filtrée au signal d'erreur codé par ondelettes. En supposant que cette erreur est calculée par rapport au signal original (schéma en boucle ouverte), cette reconstruction n'est pas optimale comme nous l'avons vu précédemment pour les pyramides laplaciennes. En effet, l'opérateur pseudo-inverse minimise la distorsion introduite dans la vidéo reconstruite en présence de bruit blanc additif sur le signal d'erreur (bande haute de la sous-bande temporelle basse). Les filtres 9/7 ayant été utilisés également dans ce schéma, il parait valide d'utiliser l'opérateur transposé comme approximation de l'opérateur pseudo-inverse pour la reconstruction. Cette modification implique simplement d'ajouter un filtrage du signal d'erreur par le filtre d'analyse H_0 au décodeur, puis de soustraire ce signal filtré après sous-échantillonnage au signal de la bande de base. Dans le schéma de codage actuel, cependant, le signal d'erreur est calculé à partir de la séquence reconstruite par le décodeur H.264. Il s'agit donc d'un schéma en boucle fermé dans lequel le signal d'erreur à coder comporte à la fois des informations de hautes fréquences perdues lors de la décimation (progressivité spatiale) et des informations de basses fréquences servant à compenser l'erreur de quantification introduite par le codeur H.264 dans la couche de base (progressivité en qualité). Le signal d'erreur ne peut donc

[4] format vidéo de résolution 352x288

[5] une progressivité en qualité est réalisée conjointement à la progressivité en résolution

[6] format vidéo de résolution 176x144

[7] apparaissant également sous la dénomination de codeur MPEG-4 AVC.

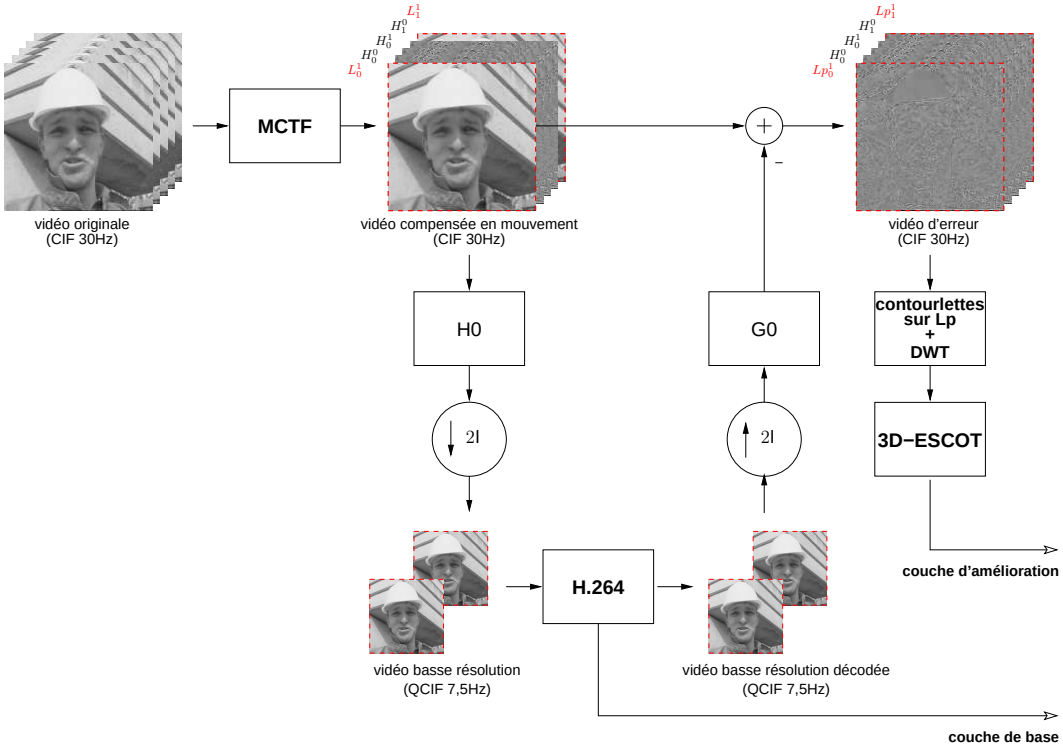

FIG. 3.19: Schéma du codeur MCWC. La séquence vidéo est tout d'abord compensée en mouvement par filtrage temporel en ondelettes. Les sous-bandes de basse fréquence temporelle sont ensuite sous-échantillonnées spatialement pour obtenir une vidéo basse résolution. Cette séquence est codée par le codeur H.264 pour obtenir le flux de la couche de base. Après décodage par H.264 cette séquence sert de prédicteur aux bandes de basse fréquence temporelle de la vidéo à pleine résolution. Cette vidéo d'amélioration est décomposée spatialement en ondelettes et les sous-bandes sont codées par 3D-ESCOT.

pas être traité directement à l'aide uniquement de la décomposition directionnelle, c'est pourquoi nous avons choisi d'utiliser une transformée en contourlettes sur 1 niveau pour séparer le signal d'erreur en ces deux composantes de nature très différente.

La deuxième modification consiste à utiliser une décomposition orientée, fournie ici par la décomposition directionnelle présentée précédemment, permettant d'obtenir 2^k sous-bandes au lieu des trois sous-bandes d'ondelettes séparables de hautes fréquences. Le choix du nombre d'orientations peut être réalisé arbitrairement ou en optimisant la distorsion pour un débit particulier. En général, un nombre d'orientations situé entre 4 et 16 est suffisant pour analyser l'information de contour du signal.

Enfin, une dernière modification du codeur est nécessaire pour adapter le codeur de sous-bandes à la nouvelle structure en contourlettes. Le codeur utilisé dans le schéma de codage vidéo MCWC est le codeur 3D-ESCOT [97], issu du codeur d'image EBCOT utilisé dans la norme JPEG2000. Ce codeur considère des voisinages temporels pour

l'estimation adaptative des lois conditionnelles de la signifiance et du signe des coefficients. Nous n'avons pas considéré la modification de ces contextes pour les adapter aux caractéristiques statistiques des contourlettes, en supposant qu'elles étaient similaires à celles des ondelettes séparables.

Pour des raisons de complexité, nous n'avons pas considéré l'application du principe de projection sur ensembles convexes dans le cadre de la compression vidéo. Cette technique nécessite en effet l'application successive de transformées et transformées inverses qu'il serait probablement trop coûteux d'effectuer dans le cadre d'une application pratique.

Nous avons appliqué ce nouveau schéma de codage à différentes séquences vidéo de résolution CIF 30Hz, chacune codée en un flux progressif. Le tableau 3.8 présente le PSNR moyen obtenu pour le codage de ces différentes séquences aux débit de 96, 192 et 256 kbits/s aux résolutions temporelles de 15Hz, 15Hz et 30Hz respectivement. Le schéma par contourlettes atteint des performances moindres que le schéma original par ondelettes d'environ 0.2 à 0.5 dB à 96kbits/s. À mesure que le débit augmente, cet écart s'accroît comme dans le cas de la compression d'images fixes. Ainsi, même pour ces images de résidu, le surcoût de codage induit par la redondance intrinsèque des contourlettes compense le gain en distorsion offert par une meilleure représentation des contours. La figure 3.20 représente le PSNR de chaque image de la séquence City pour le schéma en ondelettes et le schéma en contourlettes avec 4, 8 et 16 directions d'analyses. Les meilleures performances du schéma en contourlettes sont obtenues avec 8 directions d'analyses, qui réalisent un bon compromis entre sélectivité en directions et location des contours sur cette séquence. En effet, le banc de filtre directionnel étant à échantillonage critique, chaque coefficient correspond à une position spatiale dans l'image donnée. Ainsi, une augmentation du nombre d'orientations d'analyse induit une diminution de la résolution spatiale avec laquelle les caractéristiques de l'image correspondant à chaque orientation peuvent être localisées. La première image de résidu de la séquence City, ainsi que les images décodées par les méthodes par ondelettes et par contourlettes pour un débit de 96 kbit/s à 15Hz sont représentée sur la figure 3.21. On remarque que les informations de contours sont bien séparées des informations de correction de l'erreur de quantification par H.264, mais que les ondelettes parviennent tout de même à représenter ce signal plus efficacement en terme de compromis débit-distorsion. La figure 3.22 présente les informations conservées dans les diverses sous-bandes pour l'analyse en ondelettes et l'analyse directionnelle de cette image. Enfin, la figure 3.23 présente un exemple d'images reconstruites sur la séquence *Soccer*, montrant les performances légèrement meilleures du codeur par ondelettes.

3.7 Conclusion

Nous avons proposé de combiner la tranformée en ondelettes séparables à la transformée en contourlettes pour représenter efficacement les contours dans le cadre de la compression d'images et de vidéos. La transformée en contourlettes fournit une analyse directionnelle redondante des contours de l'image tandis que la transformée en ondelettes

séquence / schéma directions	ondelette 3	contourlette 4	8	16
City CIF 15Hz 96kbits/s	30.031	29.384	29.472	29.496
City CIF 15Hz 192kbits/s	34.108	33.030	33.150	32.993
City CIF 30Hz 256kbits/s	34.889	33.951	34.001	33.852
Mobile CIF 15Hz 96kbits/s	24.480	23.715	23.652	23.560
Mobile CIF 15Hz 192kbits/s	27.849	26.816	26.736	26.386
Mobile CIF 30Hz 256kbits/s	28.334	27.527	27.383	27.114
Soccer CIF 15Hz 96kbits/s	29.678	29.538	29.551	29.557
Soccer CIF 15Hz 192kbits/s	32.103	31.821	31.874	31.853
Soccer CIF 30Hz 256kbits/s	31.920	32.097	31.723	31.684

TAB. 3.8: Valeurs de PSNR (dB) obtenues sur la composante de luminance avec le schéma de codage vidéo par ondelettes et les différents schémas de codage vidéo par contourlettes.

permet d'obtenir une représentation compacte des variations globales de l'image. Nous avons étudié l'impact des filtres utilisés pour l'analyse multirésolution et l'analyse directionnelle en discutant des compromis réalisés pour satisfaire les objectifs de compression. Nous avons appliqué ce schéma hybride à la compression d'images en comparant la transformée en ondelettes à la transformée en contourlettes aux bas débits. Nous avons également appliqué le principe de projection sur ensembles convexes pour améliorer les performances de codage de la transformée en contourlettes. Dans le cadre d'une application concrète de ces transformées, nous avons comparé la taille des flux binaires obtenus après quantification des sous-bandes, compression par le codeur EZBC et optimisation débit-distorsion. Ces résultats montrent le potentiel des contourlettes pour la représentation des contours à bas débit. Ils sont toutefois pénalisés par la redondance de la transformée en contourlettes à haut débit. De plus, le fait de considérer un banc de filtres fixe à échantillonage critique pour l'analyse directionnelle implique de réaliser un compromis entre la précision sur la localisation spatiale des contours et la résolution en direction des filtres d'analyses. D'autre part, la sélectivité réduite du filtre 9/7 introduit une réponse non négligeable des filtres directionnels en dehors de leur domaine fréquentiel idéal. Ceci se traduit en la présence d'aliasing indésirable dans les sous-bandes directionnelles. L'ensemble de ces points contribue à la dégradation des performances en compression des contourlettes en dehors des plages de débits faibles et d'un nombre de directions d'analyse restreint.

Afin de permettre l'optimisation débit-distorsion dans le domaine transformé à l'encodage et d'obtenir une structure de décodage duale, nous nous sommes restreint à des filtres à réponse impulsionnelle finie quasi-orthogonaux. Il serait intéressant de poursuivre une étude similaire dans le cadre des filtres à réponse impulsionnelle infinie en utilisant l'opérateur pseudo inverse à la reconstruction au lieu de l'approximer par la transposée de l'opérateur d'analyse. D'autres améliorations pourraient inclure l'optimisation débit-distorsion du nombre de sous-bandes directionnelles dans le cadre d'un

codage non progressif. Il serait également possible d'adapter les contextes des codeurs de sous-bandes au cas des contourlettes, bien que le gain de codage à en attendre soit assez faible. Enfin, l'utilisation de la transformée en contourlettes CRISP [50] au lieu de la transformée en contourlettes classique permettrait de s'affranchir des problèmes liés à la redondance.

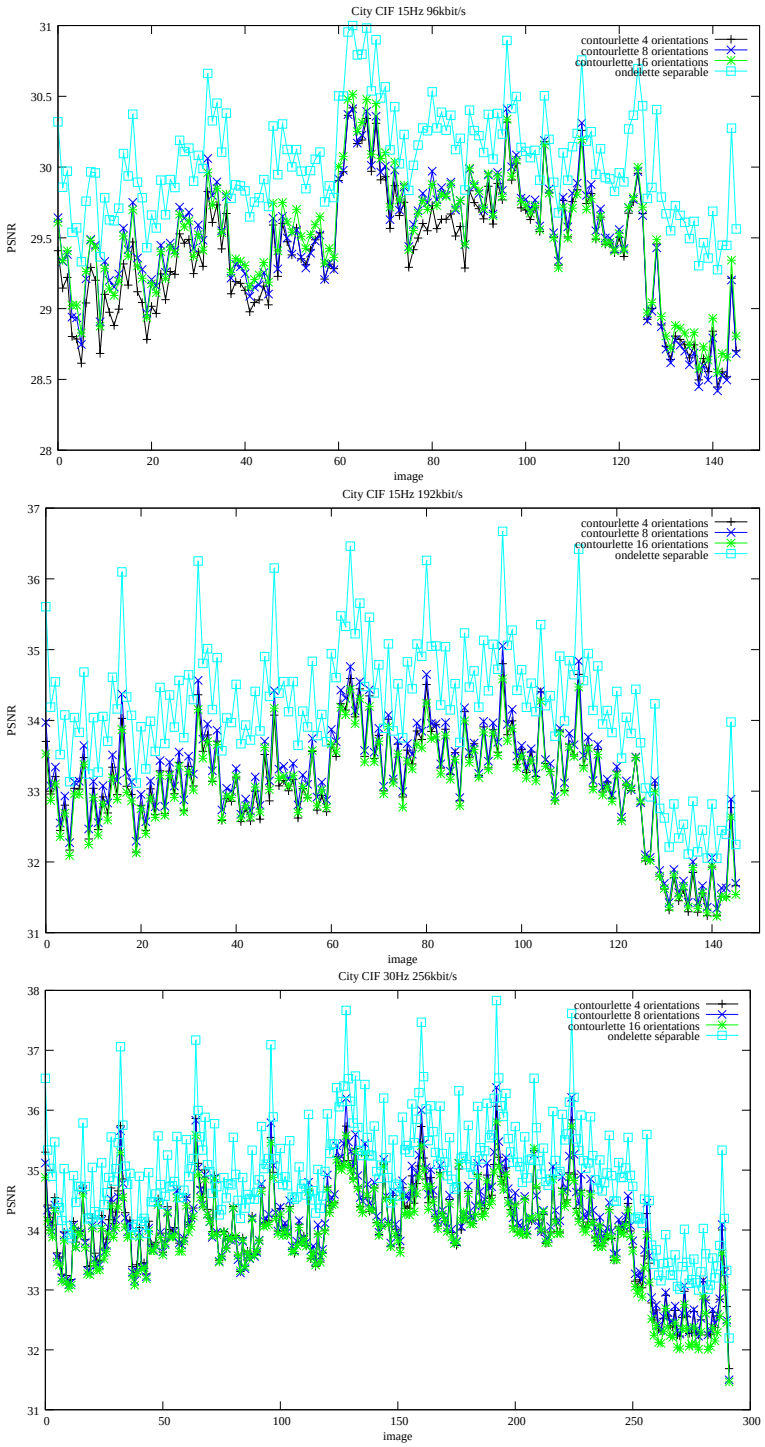

FIG. 3.20: Valeurs de PSNR obtenues sur la composante de luminance de la séquence City pour le schéma de codage vidéo par ondelettes et le schéma de codage vidéo par contourlettes à différents débits et différents nombres d'orientations.

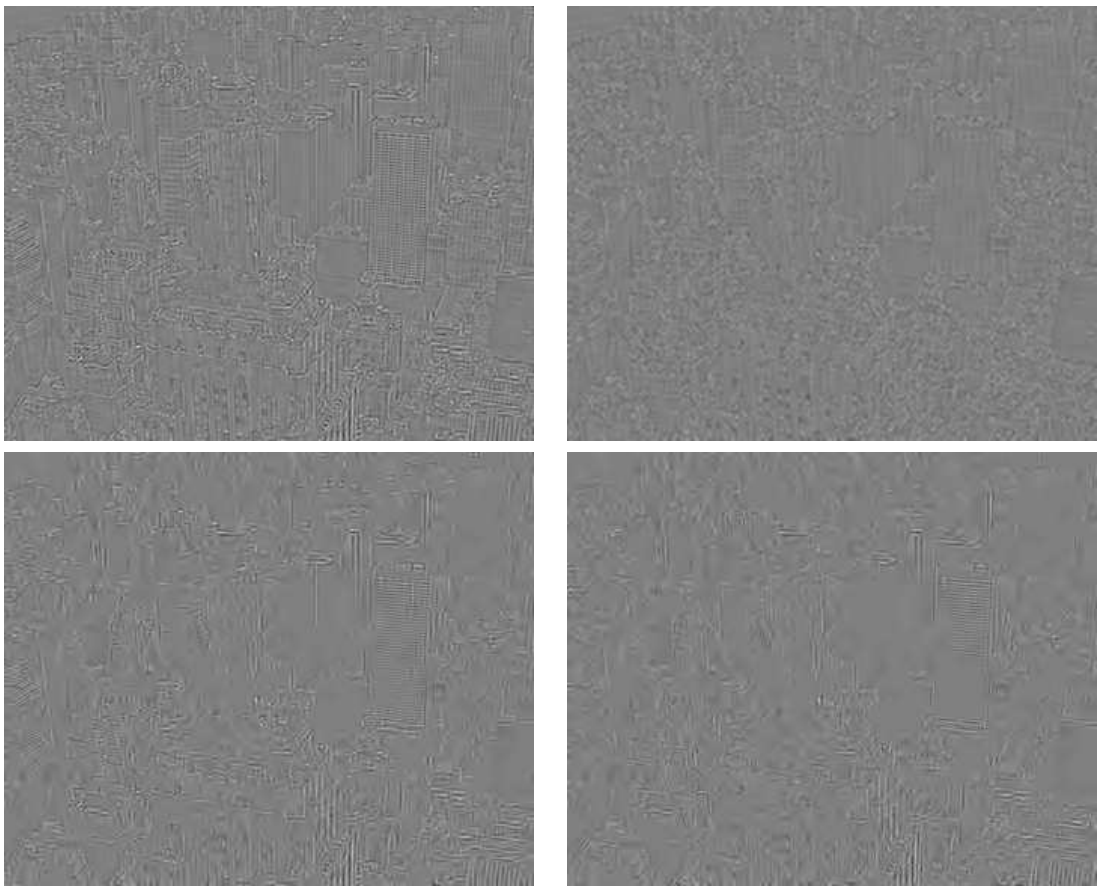

FIG. 3.21: Première image de résidu pour la séquence City. L'image originale [haut gauche] est consituée d'information de correction de l'erreur de quantification H.264 [haut droit] dans les basses fréquences et d'information de contours dans les hautes fréquences. Les images décodées pour un débit cible de 96kbit/s par le schéma par ondelettes [bas droit] et le schéma par contourlettes avec 8 directions d'analyses [bas gauche] sont représentées.

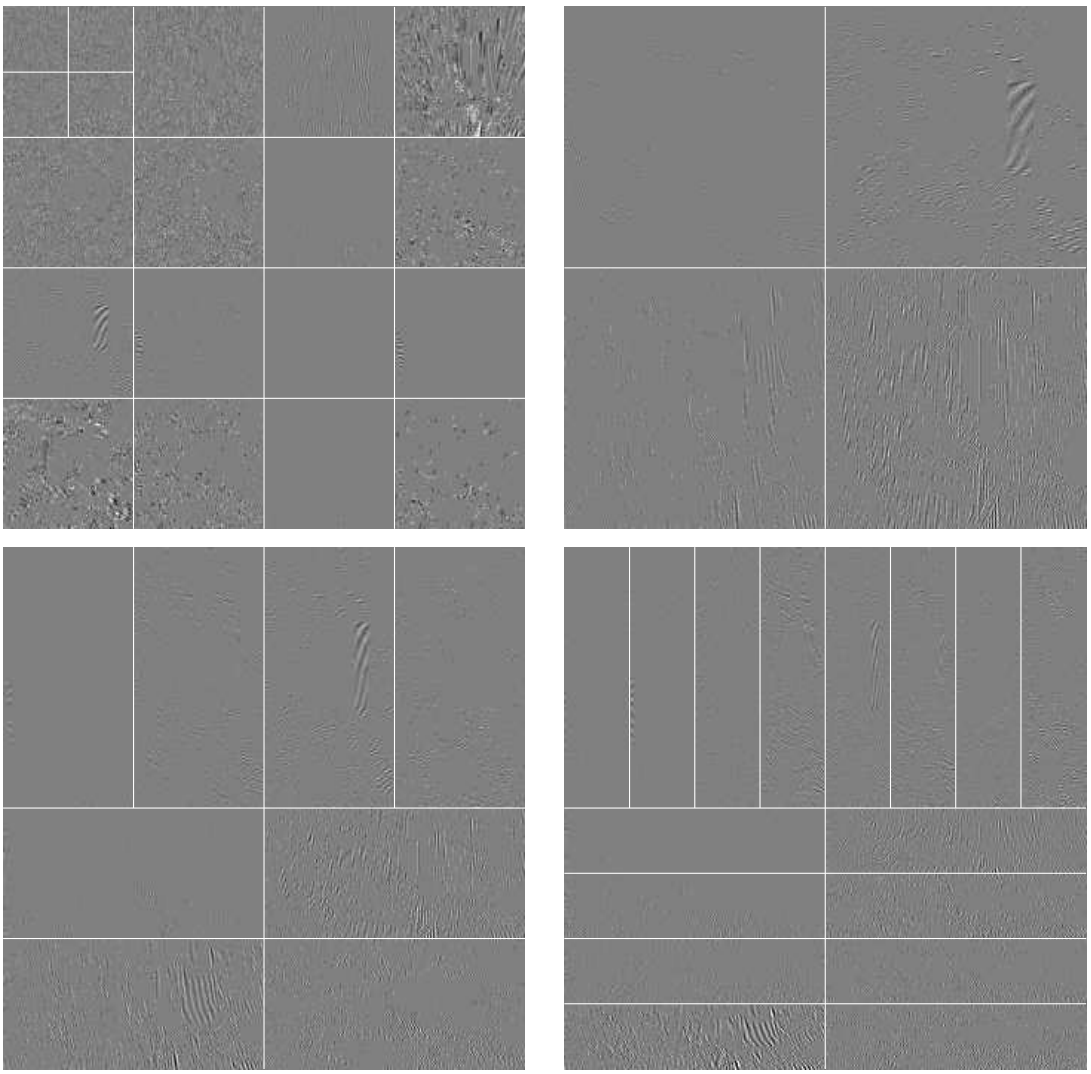

FIG. 3.22: Coefficients conservés après decodage par 3D-ESCOT dans les sous-bandes du schéma par ondelettes [haut gauche] et les sous-bandes directionnelles du schéma par contourlettes avec 4 [haut droit], 8 [bas gauche] et 16 [bas droit] directions d'analyse pour un débit cible de 192kbit/s. Dans sa configuration originale, le schéma vidéo par ondelettes effectue une décomposition supplémentaire dans les bandes de plus hautes fréquences, menant à une structure d'analyse en paquets d'ondelettes.

FIG. 3.23: Première image de la séquence Soccer CIF 15Hz décodée par le codeur vidéo en ondelettes et le codeur vidéo en contourlettes, pour un débit cible de 96kbits/s. L'image reconstruite par le codeur en ondelettes est légèrement meilleure.

Chapitre 4

Ondelettes orientées

Nous avons vu au chapitre précédent que les transformées directionnelles donnent une bonne représentation des contours des images à très bas débit. Cependant la redondance intrinsèque de la transformée en contourlettes, ainsi que la structure fixe du banc de filtres associé, ne permettent pas d'atteindre des performances satisfaisantes pour la compression à des débits plus élevés.

Dans ce chapitre, nous présentons une nouvelle transformée pour le traitement d'image basée sur les ondelettes et le cadre du lifting. Les étapes de lifting d'une ondelette unidimensionnelle sont appliqués le long d'une orientation locale définie sur une lattice quinconce. Pour maximiser la concentration d'énergie dans les basses fréquences, l'orientation minimisant l'erreur de prédiction est choisie de manière adaptative. En itérant cette décomposition sur la bande basse de l'image, on obtient une analyse multiéchelle à granularité fine.

Tant que la carte est suffisamment homogène, certaines propriétés intéressantes de l'ondelette d'origine sont préservée, comme la régularité et la quasi-orthogonalité. La reconstruction parfaite est garantie par la réversibilité des étapes de lifting.

Cette transformée s'apparente aux transformées en bandelettes et en directionlettes présentées au chapitre 2. La différence principale par rapport à ces transformées réside dans le fait de considérer des structures d'échantillonage quinconce et non pas séparables. Ceci nous permet de représenter le signal à l'aide d'une seule fonction d'ondelette. D'autre part, la transformée en bandelettes considère un choix d'orientations relativement élevé. Nous avons choisi ici de restreindre ce choix à deux orientations pour limiter le coût de codage de cette information et permettre une recherche très rapide de l'orientation locale appropriée. Il en va de même pour la transformée en directionlettes dans laquelle les directions d'analyses sont définies par les lattices de sous-échantillonage considérées, bien que nous n'ayons pas connaissance d'une application de cette transformée récente à la compression d'images.

Nous appliquons dans ce chapitre la transformée en ondelettes orientées au codage et au débruitage d'images. Dans le contexte de la compression, la carte d'orientation multirésolution est codée au moyen d'un quad-tree. L'allocation de débit entre la carte d'orientation et les coefficients d'ondelettes est optimisée conjointement en terme débit-

distorsion. Pour le débruitage, un modèle de Markov est utilisé afin d'extraire les orientations de l'image bruitée.

Nous mesurons l'information mutuelle entre les coefficients d'ondelettes et la comparons à celle observée dans le cas d'une transformée en ondelettes séparable. La performance débit-distorsion de cette nouvelle transformée est évaluée pour la compression en utilisant les codeurs de sous-bandes actuels. Dans le cadre du débruitage, les résultats obtenus avec cette nouvelle transformée sont comparés à ceux obtenus avec d'autres transformées et d'autres méthodes de débruitage.

Les ondelettes orientées se placent dans le cadre des transformées adaptatives où la géométrie de l'image est codée explicitement. Nous proposons ici un schéma dans lequel les étapes de lifting d'une ondelette unidimensionnelle, appelée *ondelette source* par la suite, sont orientées le long d'une orientation choisie parmi un ensemble fini d'orientations possibles. Ce choix est effectué de manière adaptative de sorte que l'énergie des coefficients dans les sous-bandes d'ondelettes soit minimisée. À chaque niveau de décomposition, la grille d'échantillonnage est séparée en deux classes d'équivalences complémentaires, puis les étapes lifting sont appliquées le long des orientations choisies. Cette décomposition en ondelettes orientée fournit une sous-bande haute et une sous-bande basse, sur laquelle la décomposition est itérée. Dans une application de codage d'image, la carte d'orientation multirésolution doit être transmise au décodeur pour que les étapes lifting puissent y être inversée. Dans ce cas, nous proposons de coder la carte au moyen d'un quad-tree et de réaliser une optimisation débit-distorsion pour déterminer l'équilibre entre le débit alloué à la carte et aux coefficients d'ondelettes. Dans le cadre du débruitage, nous utilisons un modèle de Markov pour la carte et un simple seuillage dans le domaine transformé. Bien que ce modèle offre une plus grande souplesse que le modèle par quad-tree, la procédure d'optimisation est plus complexe et le coût de la carte obtenue à partir d'un codage entropique contextuel reste trop élevé pour une application en compression.

La carte d'orientation sera désignée par m, et les coefficients d'ondelettes correspondants seront notés y_m. La restriction de la carte ou des coefficients d'ondelettes à un ensemble de point $\mathcal{S} \subset \mathbb{Z}^2$ sera écrite $m_\mathcal{S}$ ou $y_{\mathcal{S},m_\mathcal{S}}$ respectivement. En particulier, nous noterons $m_\mathsf{n} \in \{0,1\}$ l'orientation à la position n. En fonction de l'échelle k considérée, l'orientation m_n au point n représentera soit une orientation verticale ($m_\mathsf{n} = 0$) ou horizontale ($m_\mathsf{n} = 1$), soit une orientation diagonale ($m_\mathsf{n} = 0$) ou antidiagonale ($m_\mathsf{n} = 1$). De façon similaire, nous noterons $y_{\mathsf{n},m_\mathsf{n}} \in \mathbb{R}$ le coefficient d'ondelette au point n.

4.1 Ondelette orientée sur grille quinconce

4.1.1 Échantillonage quinconce

Considérons la lattice quinconce présentée dans l'exemple 2. La matrice

$$\mathsf{Q} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

définit une lattice $L^0_\mathcal{L} = \mathsf{Q}\mathbb{Z}^2$ et sa lattice complémentaire $L^0_\mathcal{H} = L^0_\mathcal{L} + \mathsf{j}$, avec $\mathsf{j} = (0,1)^\top$,

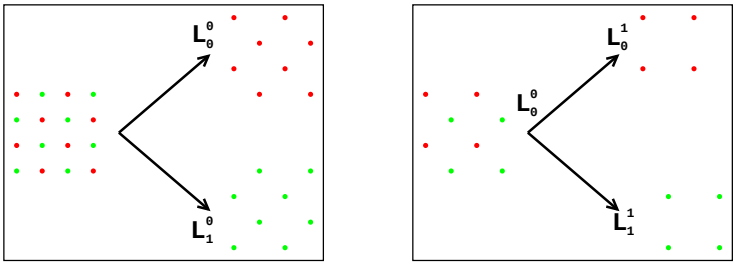

FIG. 4.1: Décomposition de la grille d'échantillonage 2D en deux lattices complémentaires.

de sorte que $\mathbb{Z}^2 = L^0_{\mathcal{L}} \cup L^0_{\mathcal{H}}$. En itérant cette décomposition sur la lattice $L^0_{\mathcal{L}}$ (Fig. 4.1), la structure multirésolution suivante est obtenue :

$$\begin{cases} L^k_{\mathcal{L}} & = L(\mathsf{Q}^{k+1}), \\ L^k_{\mathcal{H}} & = L(\mathsf{Q}^{k+1}) + \mathsf{Q}^k \mathsf{j}. \end{cases}$$

Les lattices $L^k_{\mathcal{L}}$ et $L^k_{\mathcal{H}}$ sont soit carrées soit quinconces, pour les niveaux k pairs ou impairs respectivement (en remarquant que $\mathsf{Q}^2 = 2\mathsf{I}$). Par la suite, nous appellerons *niveau quinconce* un niveau pour lequel les lattices L^k sont quinconces (k pair), tandis qu'un *niveau carré* désignera un niveau dans lequel ces lattices sont carrées (k impair). Notons cependant qu'une lattice quinconce $\mathsf{Q}^k\mathbb{Z}^2$, pour k impair, peut être vue comme une lattice carrée tournée de $\frac{\pi}{4}$ où la distance entre les échantillons est de $2^{\frac{k}{2}}$. Ainsi, cette partition récursive de $\mathbb{Z}^2$ définit une pyramide d'échantillonage quinconce $(L^0_{\mathcal{H}}, ..., L^{l-1}_{\mathcal{H}}, L^{l-1}_{\mathcal{L}})$ sur l niveaux, où le facteur de sous-échantillonnage à chaque échelle est $|\det \mathsf{Q}|^{\frac{1}{2}} = \sqrt{2}$.

Cette pyramide quinconce fournit une représentation multirésolution dont le support est identique à celui utilisé dans le cadre des ondelettes quinconces [5] [8]. De même que dans ce cadre, et contrairement au cas des ondelettes séparables, une seule fonction d'ondelette est suffisante pour représenter le signal.

4.1.2 Lifting orienté

Plutôt que d'utiliser des ondelettes quinconces, notre approche consiste à appliquer une ondelette unidimensionnelle selon une orientation sélectionnée de manière adaptative d'après une carte d'orientation. Dans cette section, nous supposons que la carte est connue et seule l'adaptation des étapes de lifting est présentée. Les sections suivantes présenteront comment obtenir la carte d'orientation en fonction de l'application.

Les coefficients d'ondelettes, correspondant aux erreurs de prédiction aux différents niveaux, sont calculés sur les lattices $L^k_{\mathcal{H}}$. Les images d'approximation sont quant à elles calculées sur les lattices $L^k_{\mathcal{L}}$ sur lesquelles la décomposition est itérée. Chaque point de $L^k_{\mathcal{H}}$ possède quatre voisins dans $L^k_{\mathcal{L}}$. Par conséquent, un point $\mathsf{n} \in L^k_{\mathcal{H}}$ peut être prédit à partir de n'importe quelle combinaison de ses voisins. Par exemple, dans le cas du lifting d'ondelettes quinconces, l'échantillon $x[\mathsf{n}]$ est prédit à partir de la moyenne de ses

quatre voisins. Ici, ce même échantillon est prédit à partir de ses voisins appartenant à la même ligne *ou* la même colonne (Fig. 4.2). Ceci nécessite la connaissance de la carte d'orientation qui définit quelle direction de prédiction est utilisée à la position n. Puisqu'il n'y a que deux choix possibles, cette carte est binaire. Généralement, l'orientation qui minimise l'erreur de prédiction est choisie, définissant ainsi la carte binaire $m_{L^k_{\mathcal{H}}}$ sur $L^k_{\mathcal{H}}$.

Pour calculer le coefficient à la position n, les étapes de prédiction de l'ondelette source sont appliquées dans l'orientation choisie en n. Ainsi, les étapes de prédiction d'une ondelette unidimensionnelle définies dans l'équation 1.7 sont modifiées, soit en l'équation :

$$P(\alpha_i) : S_1[\mathsf{n}] := S_1[\mathsf{n}] + \alpha_i \sum_{\mathsf{n}^* \in \mathcal{R}_{\mathsf{n}}} S_0[\mathsf{n}^*], \tag{4.1}$$

où $\mathcal{R}_n$ est l'ensemble des deux voisins horizontaux de n dans $L^k_{\mathcal{L}}$, soit en l'équation :

$$P(\alpha_i) : S_1[\mathsf{n}] := S_1[\mathsf{n}] + \alpha_i \sum_{\mathsf{n}^* \in \mathcal{C}_{\mathsf{n}}} S_0[\mathsf{n}^*], \tag{4.2}$$

où $\mathcal{C}_n$ est l'ensemble des deux voisins verticaux de n dans $L^k_{\mathcal{L}}$.

Notons qu'un plus grand nombre d'orientations pourrait être défini, soit en interpolant les échantillons dans $L^k_{\mathcal{L}}$ (comme cela est fait par exemple dans [104] pour les ondelettes séparables), soit en considérant des voisins plus éloignés. Toutefois, pour des raisons de simplicité et pour limiter le coût de codage de la carte, nous nous restreignons à deux orientations uniquement.

Les étapes de mise à jour sont modifiées en fonction des étapes de prédiction. Un échantillon à la position n' dans $L^k_{\mathcal{L}}$ a en effet été utilisé de zéro à quatre fois pour prédire ses voisins dans $L^k_{\mathcal{H}}$, contrairement au cas unidimensionnel où il est utilisé exactement deux fois. Les facteurs β^*_i utilisés dans les étapes de mise à jour modifiées sont obtenus en pondérant les facteurs originaux β_i de l'ondelette source donnés par l'équation 1.8. Puisque la propriété d'orthogonalité est perdue du fait de ce nombre variable de prédicteurs, le but de l'étape de mise à jour est plutôt de s'assurer que certaines propriétés statistiques du signal sont conservées dans les basses fréquences. Un autre critère pour déterminer les facteurs β^*_i adéquats aurait pu être de minimiser globalement le produit scalaire entre toutes les fonctions de base de manière à obtenir une transformée la plus orthogonale possible. Pour des raisons de complexité, cette possibilité n'a pas été considérée ici. L'approche empirique suivante, basée sur une pondération, est proposée à la place. Cette modification assure que la moyenne de l'image originale est conservée dans la bande basse dans le cas de l'ondelette 5/3. En fonction du nombre v de voisins prédits à partir d'un échantillon à la position n', les facteurs de mise à jour β^*_i sont définis par :

$$\beta^*_i = \begin{cases} \frac{2}{v}\beta_i & \text{si } v \neq 0, \\ 0 & \text{sinon.} \end{cases} .$$

Ainsi, les étapes de mises à jour sont modifiées comme suit :

$$U(\beta_i) : S_0[\mathsf{n}'] := S_0[\mathsf{n}'] + \beta_i^* \sum_{\mathsf{n}^* \in \mathcal{U}_{\mathsf{n}'}} S_1[\mathsf{n}^*], \tag{4.3}$$

où $\mathcal{U}_{\mathsf{n}'} \subset L^k_{\mathcal{H}}$ est l'ensemble des voisins de n' utilisant l'échantillon $x[\mathsf{n}']$ comme prédicteur. Quand la direction de prédiction est la même pour tous les points de $L^k_{\mathcal{H}}$, la décomposition est équivalente à l'ondelette source appliquée dans cette direction sur $L^k_{\mathcal{H}}$.

Toutes ces modifications s'appliquent de manière identique aux niveaux carrés (k impair), en les considérant comme des niveaux quinconces (k pair) tournés de $\frac{\pi}{4}$. Les orientations sont alors diagonales ou antidiagonales au lieu d'être horizontales ou verticales, mais les étapes de lifting sont appliquées de manière similaire (Fig. 4.3).

La reconstruction parfaite est garantie en inversant chaque étape de lifting, moyennant la connaissance de la carte d'orientation. Notons que le terme de *carte d'orientation* se réfère à l'ensemble des cartes multirésolution sur $L^k_{\mathcal{H}}$:

$$m = \bigcup_{k=0}^{l-1} m_{L^k_{\mathcal{H}}}.$$

De plus, si m est partitionnée en un petit nombre de régions où la direction de filtrage est constante, et si les ondelettes sources sont quasi-orthogonale, alors cette propriété est conservée pour les ondelettes orientées. En effet, dans ce cas, les ondelettes orientées sont équivalentes aux ondelettes sources appliquées sur de larges régions connexes possédant la même orientation. Par conséquent, les coefficients pour lesquels le support de l'ondelette est totalement inclut dans une région d'orientation constante ont la garantie d'être orthogonaux (ou quasi-orthogonaux) entre eux. De plus, la régularité de l'ondelette source est conservée le long de l'orientation choisie dans cette région. Seuls les coefficients aux bords de ces régions peuvent ne pas être (quasi-)orthogonaux aux autres coefficients. Ainsi, si une ondelette orthogonale est utilisée comme ondelette source, une ondelette orientée quasi-orthogonale est obtenue pour peu que la carte d'orientation soit suffisamment homogène. Cette propriété d'orthogonalité ou de quasi-orthogonalité est particulièrement importante pour les applications de compression. En effet, cette propriété permet d'effectuer l'optimisation débit-distorsion (RD) dans le domaine transformé.

Pour exploiter totalement l'information d'orientation, les sous-bandes de haute fréquence sont décomposées une nouvelle fois en deux sous-bandes (Fig. 4.4). La première contient les informations de contour tandis que la seconde contient principalement un bruit résiduel. En effet, en supposant que l'orientation est choisie de sorte à minimiser l'énergie de chaque coefficient d'ondelette dans chaque sous-bande de haute fréquence et à chaque niveau, il reste tout de même des coefficients de forte énergie correspondant aux orientations qui ne sont pas autorisées dans la carte. Comme les orientations autorisées ne sont pas les mêmes en fonction qu'il s'agisse d'un niveau quinconce ou carré,

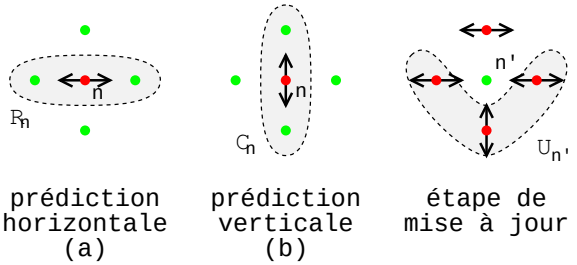

FIG. 4.2: L'étape de prédiction est appliquée soit dans la direction horizontale (a), soit dans la direction verticale (b). Les étapes de mise à jour dépendent des directions choisies pour les quatre coefficients voisins.

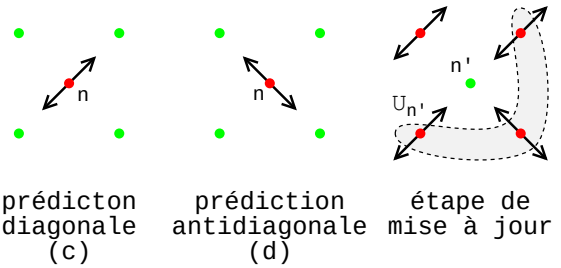

FIG. 4.3: L'étape de prédiction, dans une direction diagonale (c) ou antidiagonale (d), et l'étape de mise à jour pour les niveaux carrés.

l'information d'orientation des autres niveaux peut être utilisée pour filtrer à nouveau ces coefficients selon une orientation appropriée. Par exemple, considérons le premier niveau de décomposition. L'application du lifting orienté fournit une sous-bande de haute fréquence H0 qui a été filtrée selon des directions horizontales ou verticales, de manière à minimiser l'énergie des coefficients. Ainsi, cette sous-bande ne contient plus de basses fréquences, ni de contours horizontaux ou verticaux. En revanche, les contours diagonaux et antidiagonaux mènent à des coefficients de forte amplitude dans cette sous-bande. Cependant, puisqu'une information locale d'orientation diagonale ou antidiagonale est fournie pour effectuer la décomposition de la sous-bande H1 au niveau suivant, il est également possible d'utiliser cette même information pour décorréler la sous-bande H0 en appliquant à nouveau un lifting orienté, mais cette fois-ci selon la carte d'orientation de la sous-bande H1. On obtient alors une sous-bande H0H contenant principalement du bruit et une sous-bande H0L correspondant aux contours diagonaux et antidiagonaux, dont la taille est la moitié de la taille de la sous-bande H0. Ainsi, l'énergie de l'image est concentrée encore davantage. La figure 4.4 illustre le processus complet de décomposition.

La figure 4.5 représente les coefficients d'une décomposition en ondelettes séparables et en ondelettes orientées de la même image *zoneplate*. Les sous-bandes quinconces de

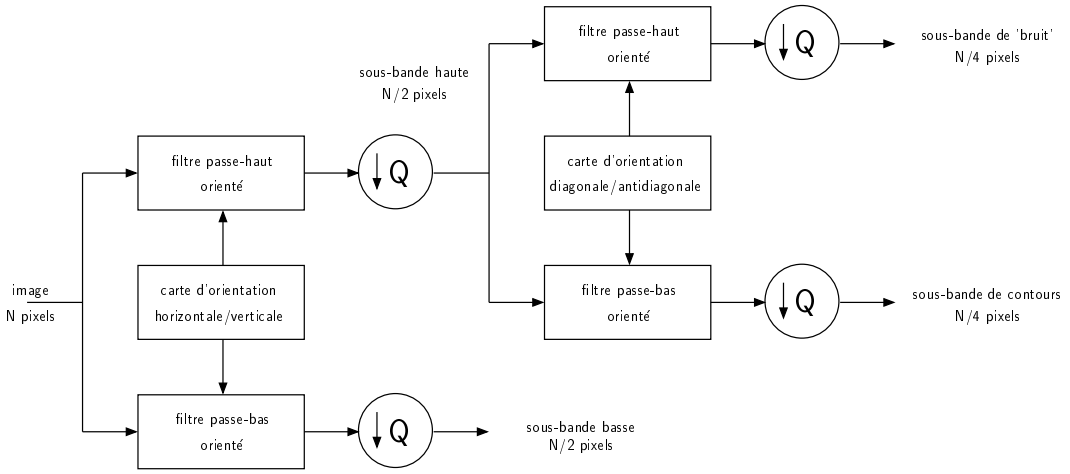

FIG. 4.4: Un niveau de décomposition en sous-bandes orientées. L'image d'origine est séparée en une sous-bande de basse fréquence et une sous-bande de haute fréquence à l'aide d'une carte d'orientation. Cette décomposition est itérée sur la sous-bande basse, tandis que la sous-bande haute est à nouveau décomposée en une sous-bande de contours et une sous-bande de bruit d'après la carte d'orientation complémentaire.

la décomposition en ondelettes orientées sont condensées horizontalement en un rectangle pour une représentation plus aisée. Les sous-parties du plan fréquentiel conservées dans chaque sous-bande apparaissent clairement pour les deux transformées. Les cartes d'orientations horizontale/verticale et diagonale/antidiagonale utilisées pour le filtrage en ondelettes orientées sont également représentées. Ces cartes ont été obtenues en codant l'image *zoneplate* au débit de 0.5 bpp avec le codeur présenté par la suite.

4.2 Application à la compression d'image

4.2.1 Représentation de la carte d'orientation par Quad-Trees

Afin de concentrer le plus possible l'énergie de l'image dans les sous-bandes de basses fréquences, la carte d'orientation est choisie de manière à minimiser l'erreur de prédiction à chaque niveau. Sans codage entropique, cette carte d'orientation coûterait à peine moins d'un bit par pixel, ce qui est un coût prohibitif par rapport au coût des coefficients d'ondelettes. Cependant, cette information binaire sur la direction de filtrage n'est pas toujours significative. En effet, lorsque la distorsion obtenue par la prédiction dans l'une ou l'autre des directions est similaire, le choix de l'orientation appropriée n'impacte pas significativement la distorsion. Ceci apparaît principalement dans les zones uniformes de l'image, où les deux prédicteurs sont similaires, ainsi que dans les zones texturées, où les pixels sont moins corrélés et ne permettent pas d'obtenir de bonnes prédictions quelle que soit la direction choisie. Ainsi, l'information d'orientation est uniquement importante sur les contours de l'image, qui constituent une proportion assez faible des

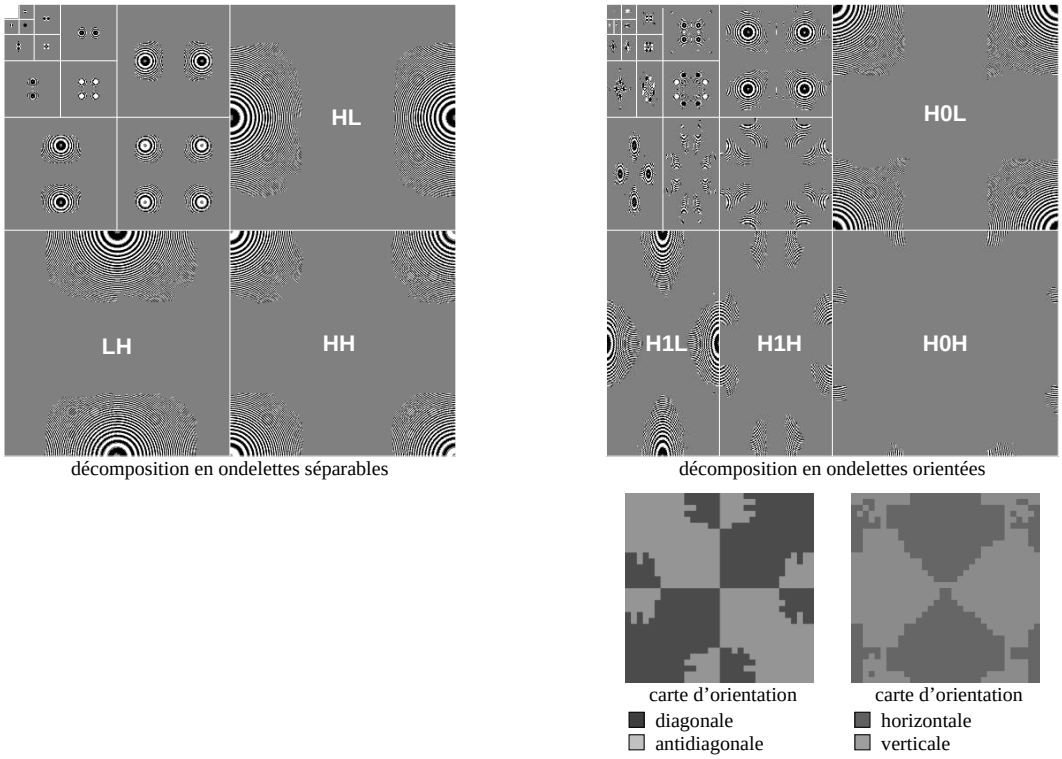

FIG. 4.5: Coefficients conservés par EBCOT pour une décomposition en ondelettes séparables [gauche] et orientées [droite] de l'image *zoneplate* pour un débit cible de 0.5 bpp. Les PSNR des images reconstruites correspondantes sont respectivement de 19.2 dB et 20.6 dB. Les cartes d'orientations correspondantes pour la transformée en ondelettes orientées sont également représentées.

pixels dans les images naturelles. Il est donc possible de propager l'information d'orientation des contours vers les autres zones de l'image de manière à réduire de façon conséquente l'entropie de la carte tout en ayant un impact négligeable sur la distorsion globale.

Pour faire cela, la carte d'orientation est codée en utilisant deux quad-trees indépendants (Fig. 4.6). L'un est utilisé pour coder les orientations horizontales et verticales (Fig. 4.2) des niveaux pairs, tandis que l'autre est utilisé pour coder les orientations diagonales et antidiagonales (Fig. 4.3) des niveaux impairs. La relation parent-enfant dans ces quad-trees permet de plus d'extraire la dépendance forte entre les orientations aux différents niveaux de l'arbre.

Au premier niveau de décomposition (niveau 0), les coefficients d'ondelettes sont définis sur une lattice quinconce et orientés soit verticalement soit horizontalement (Fig. 4.2). Afin de définir un quad-tree $\mathcal{Q}_{HV}$ représentant ces orientations sur une grille quinconce, les coefficients d'ondelettes aux niveaux quinconces sont groupés en paires partageant la même orientation (Fig. 4.7A). Puisque la séparation de la carte jusqu'au

niveau d'un coefficient serait trop coûteuse en débit de toute manière, cette contrainte n'a qu'un impact négligeable sur la procédure globale d'optimisation. Au second niveau de décomposition (niveau 1), les coefficients d'ondelettes sont définis sur une lattice carrée et possèdent une orientation soit diagonale soit antidiagonale. Un autre quad-tree $\mathcal{Q}_{DA}$ est utilisé pour représenter ces orientations, où chaque coefficient est associé directement à un noeud feuille de ce quad-tree (Fig. 4.7B).

Un sous-arbre $\mathcal{T}_{\mathsf{n}}$ d'un de ces quad-trees $\mathcal{Q}$ se définit comme le quad-tree dont le noeud racine est $\mathsf{n} \in \mathcal{Q}$ et qui contient tous les fils de n dans $\mathcal{Q}$.

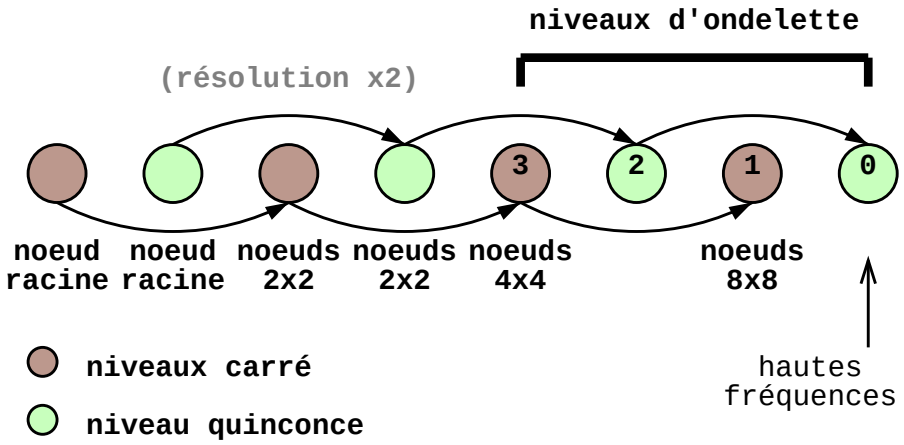

FIG. 4.6: Dépendance inter-échelle des quad-trees pour une décomposition sur 8 niveaux d'une image 16x16.

4.2.2 Procédure d'optimisation et de régularisation

L'erreur de prédiction à la position n correspond à un coefficient d'ondelette $y_{\mathsf{n},m_{\mathsf{n}}}$. Ce coefficient est en principe obtenu en appliquant le filtre d'ondelette passe-haut à la position n. Ce filtre adaptatif et non séparable dépend non seulement de l'orientation m_{n} choisie en n mais également de l'orientation choisie dans un voisinage de n dont la taille dépend du support de l'ondelette source. Par conséquent, une procédure d'optimisation globale est requise pour minimiser l'énergie des coefficients globalement. Ici, pour des raisons de simplicité et d'efficacité de calcul, une simple interpolation linéaire est utilisée afin de calculer les prédicteurs pendant l'optimisation de la carte. Le coefficient d'ondelette $y_{\mathsf{n},m_{\mathsf{n}}}$ à la position n pour l'orientation m_{n} est approximé par l'erreur de prédiction suivante :

$$e_{\mathsf{n},m_{\mathsf{n}}} = \begin{cases} S[\mathsf{n}] - \frac{1}{2} \sum\limits_{\mathsf{n}^* \in \mathcal{R}_{\mathsf{n}}} S[\mathsf{n}^*] & \text{si } m_{\mathsf{n}} = 0, \\ S[\mathsf{n}] - \frac{1}{2} \sum\limits_{\mathsf{n}^* \in \mathcal{C}_{\mathsf{n}}} S[\mathsf{n}^*] & \text{si } m_{\mathsf{n}} = 1. \end{cases}$$

Cette expression ne dépend que de l'orientation m_{n} choisie en n, ce qui permet une optimisation plus simple dans la mesure où elle est locale. Notons que cette erreur

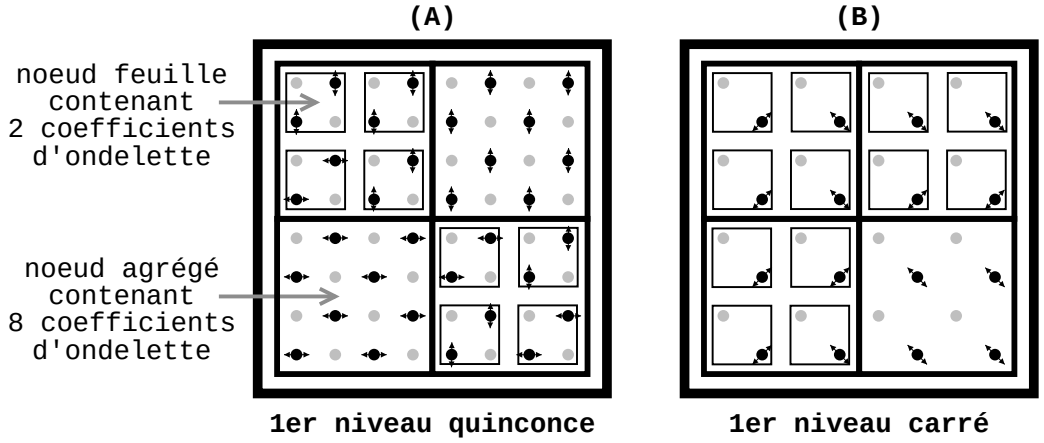

FIG. 4.7: Position des coefficients d'ondelettes pour le premier (A) et le second (B) niveau de décomposition. Les coefficients de basse fréquence correspondants sont grisés. Les noeuds des quad-trees sont représentés par des carrés d'épaisseur variable, où le noeud racine est représenté par le carré externe le plus épais.

correspond à la valeur exacte du coefficient d'ondelette dans le cas de l'ondelette 5/3. Plutôt que de calculer le débit et la distorsion exacte pour chaque configuration, ce qui serait trop coûteux en temps de calcul, l'approximation bas débit proposée dans [84] est utilisée. Ce modèle approxime la quantification par un seuillage et à considérer que la contribution au débit de chaque coefficient codé est fixe et indépendante. Le débit R et la distorsion D sont approximés par :

$$R(\mathsf{n}, m_{\mathsf{n}}) = \begin{cases} \gamma_0 & \text{si } e^2_{\mathsf{n},m_{\mathsf{n}}} > \sigma \\ 0 & \text{sinon,} \end{cases}$$
$$D(\mathsf{n}, m_{\mathsf{n}}) = \begin{cases} 0 & \text{si } e^2_{\mathsf{n},m_{\mathsf{n}}} > \sigma \\ e^2_{\mathsf{n},m_{\mathsf{n}}} & \text{sinon,} \end{cases}$$

où $\sigma = \frac{4}{3}\lambda\gamma_0$ et γ_0 approxime le coût en bits d'un coefficient codé. Ici, la valeur $\gamma_0 = 6$ est choisie. Ce modèle approxime correctement la courbe débit-distorsion réelle des coefficients d'ondelettes orientées aux bas débits.

La carte est tout d'abord initialisée avec les orientations qui minimisent l'erreur de prédiction, ce qui mène à une carte très hétérogène. Le débit $R(\mathsf{n}, m_{\mathsf{n}})$ et la distorsion $D(\mathsf{n}, m_{\mathsf{n}})$ associés à chaque noeud n de l'arbre sont calculés pour les deux orientations en utilisant le modèle ci-dessus. Notons $\mathcal{T}_{\mathsf{n}}$ le sous-arbre de noeud racine n dans le quad-tree. Alors, étant donné un paramètre λ variable, la carte est optimisée par approche ascendante en calculant les lagrangiens :

$$J_{merge}(\mathsf{n}) = \min_{m_\mathsf{n}\in\{0,1\}}(\sum_{\mathsf{f}\in\mathcal{T}_\mathsf{n}} D(\mathsf{f}, m_\mathsf{n}) + \lambda(\sum_{\mathsf{f}\in\mathcal{T}_\mathsf{n}} R(\mathsf{f}, m_\mathsf{n}) + R_\mathsf{n})),$$
$$J_{split}(\mathsf{n}) = \min_{m_{\mathcal{T}_\mathsf{n}}}(\sum_{\mathsf{f}\in\mathcal{T}_\mathsf{n}} D(\mathsf{f}, m_\mathsf{f}) + \lambda(\sum_{\mathsf{f}\in\mathcal{T}_\mathsf{n}} R(\mathsf{f}, m_\mathsf{f}) + R_{\mathcal{T}_\mathsf{n}})).$$

La même orientation est utilisée pour tous les noeuds du sous-arbre $\mathcal{T}_\mathsf{n}$ si $J_{merge}(\mathsf{n}) < J_{split}(\mathsf{n})$. Le débit R_n est le coût introduit par le codage de l'orientation en n, et est fixé à 1. Le débit $R_{\mathcal{T}_\mathsf{n}}$ représente le coût du sous-arbre $\mathcal{T}_\mathsf{n}$. Cette procédure d'optimisation est répétée en faisant varier λ, jusqu'à ce que le débit total désiré soit atteint. Le paramètre λ correspondant est obtenu par un algorithme de descente du gradient sur $R(\lambda)$.

Pour réduire la complexité de la procédure d'optimisation et pour conserver de bonnes propriétés de quasi-orthogonalité des ondelettes orientées, la résolution de la carte d'orientation est limitée à des blocs de 16x16 pixels. Ceci correspond à agréger les quatre derniers niveaux des quad-trees.

Exemple 9:

Par souci de clarté observons la procédure d'optimisation sur un arbre binaire $\mathcal{Q}$ plutôt qu'un quad-tree (Fig. 4.8). Considérons les noeuds a, b, c, d *et* e *de l'arbre. Les noeuds* a *et* b *sont des feuilles voisines partageant le même parent c. Par conséquent, le sous-arbre $\mathcal{T}_\mathsf{c}$ dans $\mathcal{Q}$ est l'ensemble $\{\mathsf{c}, \mathsf{a}, \mathsf{b}\}$. Le noeud* d *illustre le cas d'un noeud agrégé. Le noeud* e *est un noeud interne possédant une orientation différente des ses fils, qui partagent la même orientation.*

L'optimisation se réalise de la façon suivante. Considérons la feuille à la position a. *Soient $J(\mathsf{a}, 0)$ et $J(\mathsf{a}, 1)$ les fonctions de coût à la position* a *pour l'orientation $m_\mathsf{a} = 0$ et $m_\mathsf{a} = 1$ respectivement :*

$$J(\mathsf{a}, 0) := D(\mathsf{a}, 0) + \lambda R(\mathsf{a}, 0),$$
$$J(\mathsf{a}, 1) := D(\mathsf{a}, 1) + \lambda R(\mathsf{a}, 1),$$
$$J(\mathsf{a}, \star) := \min(J(\mathsf{a}, 0), J(\mathsf{a}, 1)).$$

Par exemple, supposons que $J(\mathsf{a}, \star) = J(\mathsf{a}, 0)$, alors l'orientation 0 est choisie à la position a. *Le point débit-distorsion optimal pour le sous-arbre $\mathcal{T}_\mathsf{a} = \{\mathsf{a}\}$ est stocké dans $J(\mathcal{T}_\mathsf{a}, \star)$. De plus, le point débit-distorsion correspondant au choix de la même orientation 0 pour tous les noeuds du sous-arbre $\mathcal{T}_\mathsf{a}$ est stocké dans $J(\mathcal{T}_\mathsf{a}, 0)$. Enfin, de manière similaire, le point débit-distorsion correspondant au choix de l'orientation 1 pour tous les noeuds du sous-arbre $\mathcal{T}_\mathsf{a}$ est stocké dans $J(\mathcal{T}_\mathsf{a}, 1)$.*

En suivant la même procédure pour le noeud b *que celle décrite pour le noeud* a, *imaginons que l'orientation 1 soit choisie à la position* b. *A la position* c, *le point débit-distorsion optimal $J(\mathsf{c}, \star)$ est obtenu par la même procédure. Les points débit-distorsion $J(\mathcal{T}_\mathsf{c}, 0)$ et $J(\mathcal{T}_\mathsf{c}, 1)$ correspondant au choix de la même orientation 0 ou 1 respectivement pour le sous-arbre $\mathcal{T}_\mathsf{c} = \{\mathsf{c}\} \cup \mathcal{T}_\mathsf{a} \cup \mathcal{T}_\mathsf{b}$ sont mis à jour :*

$$J(\mathcal{T}_{\mathsf{c}},0) = J(\mathsf{c},0) + J(\mathcal{T}_{\mathsf{a}},0) + J(\mathcal{T}_{\mathsf{b}},0),$$
$$J(\mathcal{T}_{\mathsf{c}},1) = J(\mathsf{c},1) + J(\mathcal{T}_{\mathsf{a}},1) + J(\mathcal{T}_{\mathsf{b}},1).$$

L'algorithme décide ensuite d'agréger ou non les noeuds a, b *et* c *en calculant et comparant*

$$\begin{aligned} J_{split}(\mathsf{c}) &= J(\mathcal{T}_{\mathsf{a}},\star) + J(\mathcal{T}_{\mathsf{b}},\star) + J(\mathsf{c},\star) + \lambda \\ &= J(\mathsf{a},0) + J(\mathsf{b},1) + J(\mathsf{c},\star) + \lambda \\ J_{merge}(\mathsf{c}) &= \min(J(\mathcal{T}_{\mathsf{c}},0), J(\mathcal{T}_{\mathsf{c}},1)) + \lambda \\ &= \min(J(\mathsf{a},0) + J(\mathsf{b},0) + J(\mathsf{c},0), \\ &\qquad J(\mathsf{a},1) + J(\mathsf{b},1) + J(\mathsf{c},1)) + \lambda \end{aligned}$$

En supposant $J_{split}(\mathsf{c}) < J_{merge}(\mathsf{c})$*, le noeud n'est pas agrégé. Le point débit-distorsion optimal est alors mis à jour :*

$$J(\mathcal{T}_{\mathsf{c}},\star) = J(\mathsf{c},\star) + J(\mathsf{a},\star) + J(\mathsf{b},\star) + \lambda.$$

Si en revanche $J_{split}(\mathsf{n}) > J_{merge}(\mathsf{n})$ *pour un noeud* n*, ce noeud est agrégé, comme l'illustre la figure 4.8 à la position* d*. La même orientation qu'en* d *est alors adoptée pour tous les noeuds fils, et les points débit-distorsion sont mis à jour (en supposant pour l'exemple que l'orientation* 0 *est choisie) :*

$$J(\mathcal{T}_{\mathsf{d}},\star) = J(\mathcal{T}_{\mathsf{d}},0).$$

La position e *illustre un cas où le noeud parent* e *est orienté différemment de ses fils. Bien que les deux noeuds fils partagent la même orientation, ce noeud ne peut être agrégé.*

4.2.3 Complexité

En utilisant les approximations présentées dans la section précédente, la complexité de la transformée en ondelette orientées, incluant l'estimation de la carte, reste comparable à celle de la transformée en ondelettes séparables. Pour une ondelette comportant K étapes de lifting (y compris les deux dernières étapes de normalisation), le nombre total d'opérations de multiplication pour une décomposition sur un niveau d'une image de taille SxS à l'aide d'ondelettes séparables est donné par KS^2. Comme cette décomposition est itérée sur la sous-bande basse de taille $\frac{1}{4}S^2$, le nombre total d'opérations de multiplication pour une décomposition sur L niveaux est donné par $(1-\frac{1}{4^L})\frac{4}{3}KS^2$.

Dans le cas de la transformée en ondelettes orientées, un premier filtrage est effectué pour obtenir les sous-bandes de haute et de basse fréquence. Ensuite, excepté

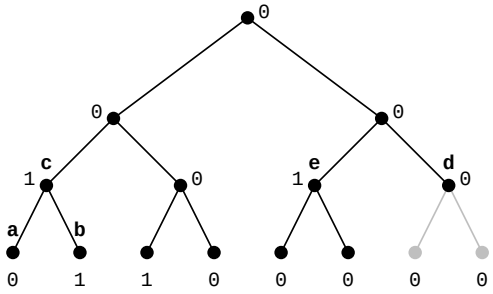

FIG. 4.8: Illustration de la procédure d'optimisation sur un arbre binaire (Ex. 9). Une approche ascendante est utilisée, partant des feuilles de l'arbre, comme les noeuds a et b, jusqu'au noeud racine. Le noeud c n'est pas agrégé, permettant au noeuds a et b d'adopter des orientations différentes de celle de c. Le noeud d représente un noeud agrégé. Le noeud e possède une orientation différente de ses fils, et n'est par conséquent pas agrégé.

pour le dernier niveau de décomposition, un second filtrage est effectué sur la sous-bande de haute fréquence. Il en résulte soit $\frac{1}{2}KS^2$ opérations de multiplication pour le dernier niveau, soit $\frac{3}{4}KS^2$ pour les autres niveaux. Ainsi, pour une décomposition sur L niveaux de décomposition, le nombre total d'opérations de multiplication est donné par $(1 - \frac{4}{3}\frac{1}{2^L})\frac{3}{2}KS^2$. Asymptotiquement, la transformée en ondelettes orientée est donc légèrement plus complexe que la transformée en ondelettes séparables, avec $1,5K$ opérations de multiplication par pixel contre $1,33K$. Pour une résolution donnée de la sous-bande de basse fréquence, le nombre de décompositions pour une transformée en ondelette orientée est double de celui d'une transformée en ondelettes séparables. Dans ce cas, le nombre total d'opérations de multiplication par pixel est supérieur d'au plus $12,5\%$ pour la transformée en ondelettes orientées par rapport à la transformée en ondelettes séparables.

L'estimation de l'orientation est de complexité réduite par rapport aux opérations de filtrage, sauf pour des ondelettes simples permettant de calculer la transformée correspondante sans multiplication, comme par exemple les ondelettes 5/3. En effet, à chaque niveau de décomposition, l'orientation est choisie en comparant l'énergie de l'erreur de prédiction pour deux orientations, sur des blocs de taille 16x16. Les erreurs de prédiction sont obtenues en utilisant uniquement des opérations d'addition et de décalage, et sont finalement élevées au carré pour obtenir la distorsion. Il en résulte un total de $2(1 - \frac{1}{2^L})$ multiplications par pixel. La complexité pourrait être encore réduite en utilisant des sommes de différences absolues comme approximation de l'énergie, bien que ceci affecte les performances et ne soit pas considéré dans l'approche présentée ici.

Finalement, il est important de noter que l'ensemble des opérations de lifting peuvent s'effectuer sans mémoire supplémentaire. Les besoins en mémoire pour la transformée en ondelettes orientée sont par conséquent très proches de ceux de la transformée en

ondelettes séparables, car la carte d'orientation est binaire et limitée à des blocs de taille fixe.

4.2.4 Codage de la carte d'orientation

Pour chaque quad-tree, l'information d'orientation est codée en partant de la racine vers les feuilles. Chaque noeud est codé en utilisant un bit pour spécifier son orientation (appelé bit d'*orientation*) et un bit pour spécifier si il a des fils ou non (appelé bit de *descendance*). Les noeuds d'un sous-arbre dont la racine indique qu'elle n'a pas de fils ne sont pas considérés. Le bit de *descendance* est compressé à l'aide d'un codeur arithmétique binaire adaptatif. Ceci permet de réduire la taille de ce flux de l'ordre de 10% pour une complexité légèrement accrue et reste optionnel. Le bit d'*orientation* est représenté directement.

Bien que ce codage soit extrêmement simple et rapide, le débit de la carte d'orientation reste négligeable comparé au débit des coefficients d'ondelettes. La compression de cette information pourrait être améliorée en utilisant des codeurs contextuels, aux dépends d'une complexité accrue. Notons également que pour améliorer la résistance de la carte d'orientation aux erreurs de transmission, tous les noeuds de la carte peuvent être codés à l'aide d'un seul bit indiquant leur orientation. Pour une taille de bloc de 16x16 pixels, on obtient un code à longueur fixe de taux 0.01 bpp pour une image typique de taille 512x512.

4.2.5 Codage des coefficients d'ondelettes

Afin d'évaluer la performance de la transformée en ondelettes orientées dans le contexte de la compression d'images, les coefficients doivent être codés efficacement. L'ondelette 9/7 est utilisée, à la fois comme référence d'ondelette séparable et comme ondelette source des ondelettes orientées. En première approximation, une simple quantification par zone morte est effectuée sur toutes les sous-bandes et le débit est estimé en calculant l'entropie absolue, la loi des coefficients quantifiés étant estimée au moyen d'un histogramme par sous-bande. L'image reconstruite correspond à celle qui serait obtenue avec un codeur non progressif très simple supposant les coefficients indépendants. Puisque les ondelettes orientées ont de meilleures propriétés de décorrélation que les ondelettes séparables, un gain significatif en termes de PSNR est observé sur une large plage de débits, en particulier pour les images contenant des caractéristiques directionnelles (Fig. 4.18). L'image reconstruite contient également moins d'artefacts visuels car les contours sont mieux préservés par la transformée en ondelettes orientée.

Le tableau 4.1 compare les performances en PSNR de la transformée en ondelettes orientées par rapport à la transformée en ondelettes séparables à entropie absolue constante. Un gain de 0.5dB à 1.3dB est observé sur l'ensemble des images de test à une entropie de 0.3 bpp. Bien que les techniques basées sur les contourlettes présentées au chapitre 3 donnent des performances intéressantes sur les images à fort contenu directionnel, elles n'offrent pas de résultats satisfaisant comparé aux ondelettes orientées dans le cas général. Les auteurs de [84] annoncent un gain comparable de 0.5 dB à

1.5 dB pour la transformée en bandelettes par rapport à la transformée en ondelettes séparables, en utilisant un codeur arithmétique adaptatif sur les images *lena* et *barbara* respectivement. Nous obtenons approximativement la même performance avec la transformée en ondelettes orientées, avec toutefois une complexité réduite et un codeur non adaptatif.

image	lena	barbara	mandrill	bike
separable (dB)	33.98	26.93	23.26	23.66
orientée (dB)	34.73	28.26	23.75	24.55
carte (%)	1.5%	1.8%	2.2%	2.9%

TAB. 4.1: Comparaison des performance en PSNR de la transformée en ondelettes séparables et orientées pour une entropie absolue de 0.3bpp et une décomposition sur 3 et 6 niveaux respectivement. La proportion du débit de la carte par rapport au débit total est également indiquée.

Bien qu'elle donne une bonne idée générale de la performance en compression de la transformée en ondelettes orientées, l'entropie absolue n'est pas une mesure suffisante. En effet, elle ne représente pas de manière très fidèle les performances des codeurs de sous-bandes actuels, qui tiennent compte des dépendances d'ordre supérieur entre les coefficients au moyen de modèles contextuels.

Pour évaluer les performances débit-distorsion dans un système réel de compression, nous avons utilisé le codeur entropique EBCOT de JPEG2000[1] [19] à la transformée ondelette orientée et comparé le débit résultant (incluant le coût de la carte d'orientation) avec celui obtenu après utilisation de la transformée en ondelettes séparable. Les sous-bandes d'ondelettes orientées sont vues par le codeur EBCOT comme des sous-bandes diagonales, sans autre modification plus avancée. Notons qu'on pourrait s'attendre à de meilleures performances si les modèles contextuels étaient adaptés aux statistiques des sous-bandes d'ondelettes orientées. Bien que l'adaptation utilisée soit sous-optimale, un léger gain est obtenu en général par rapport à la transformée en ondelettes séparables. Par exemple, un gain de 0.3 dB est observé sur l'image *barbara* à un débit de 0.3 bpp (Table 4.2). Cependant, la transformée en ondelettes orientées obtient des performances très légèrement inférieures à la transformée en ondelettes séparables sur l'image *bike* avec EBCOT, bien qu'un gain significatif est observé lorsque l'entropie absolue est utilisée comme estimateur du débit. Les courbes illustrées aux figures 4.19, 4.20 et 4.21 montrent les performances objectives de la transformée en ondelettes séparables et de la transformée en ondelettes orientées couplées au codeur EBCOT pour diverses images. Les performances des deux transformées sont similaires sur une large plage de débits, les ondelettes orientées offrant de meilleurs performances à bas débit. La courbe entropie-distorsion illustrée figure 4.18 montre le gain apporté par la transformée en ondelettes orientées par rapport à la transformée en ondelettes séparables lorsque les coefficients sont supposés indépendants. Les figures 4.18 et 4.19 soulignent également le

[1] extrait de OpenJpeg 0.95 (`http ://www.openjpeg.org`)

rôle extrêmement important du codeur de sous-bande EBCOT pour obtenir de bonnes performances avec la transformée en ondelettes séparables.

Les figures 4.9, 4.11, 4.13, 4.15 et 4.17 permettent d'observer la qualité des images reconstruites après codage par EBCOT pour les transformées par ondelettes séparables et par ondelettes orientées. De manière générale, les contours sont mieux préservés et bien localisés par les ondelettes orientées. L'effet de pixelisation des ondelettes séparables n'apparaît pas avec les ondelettes orientées, en particulier dans les régions texturées. Certaines zones uniformes sont cependant parfois moins bien préservées par les ondelettes orientées lorsqu'un changement d'orientation à lieu dans ces régions. À qualité objective équivalente, les images paraissent néanmoins de qualité visuelle légèrement meilleure avec les ondelettes orientées.

image	lena	barbara	mandrill	bike
separable (dB)	34.93	29.17	23.63	25.07
orientée (dB)	35.05	29.52	23.90	25.06

TAB. 4.2: Comparaison des performances en PSNR de la transformée en ondelettes séparables et en ondelettes orientées à un débit de 0.3 bpp en utilisant EBCOT pour une décomposition en 3 et 6 niveaux respectivement.

Généralement, le gain observé avec EBCOT est bien plus faible que lorsque le débit est estimé par l'entropie absolue. Ceci n'est pas surprenant étant donné que le codeur EBCOT exploite efficacement la dépendance résiduelle entre les coefficients d'ondelettes séparables. Comme cette dépendance est plus faible pour les coefficients d'ondelettes orientées, le gain obtenu par l'utilisation de modèles contextuels d'ordre supérieur des codeurs de sous-bande actuels est réduit. Par conséquent, l'écart de performance débit-distorsion entre la transformée en ondelettes séparables et la transformée en ondelettes orientées est réduit. Un comportement similaire est attendu des autres transformées orientées, comme la transformée en bandelettes, bien qu'à notre connaissance aucune expérience similaire n'ait été conduite pour ces transformées.

En comparant le tableau 4.1 au tableau 4.2, on remarque que la transformée en ondelettes orientées reposant sur un codeur de sous-bande basique obtient des performances à moins d' 1 dB des performances de la transformée en ondelettes séparables utilisant un codeur de sous-bandes complexe comme EBCOT. Ceci suggère qu'un codeur entropique simple, à complexité réduite, pourrait être suffisant pour obtenir des performances de compression acceptables, tout en autorisant un décodage partiel très rapide. Ceci peut être d'un grand intérêt pour les applications capables d'effectuer des traitements directement dans le domaine transformé.

FIG. 4.9: [haut] Image *barbara* 512x512 codée en utilisant 5 niveaux de décomposition d'ondelettes 9/7 séparables et le codeur EBCOT à 0.3 bpp, PSNR = 29.27 dB. [bas] Même image codée en utilisant 10 niveaux d'ondelettes 9/7 orientées et le codeur EBCOT à 0.3 bpp, PSNR = 29.61 dB.

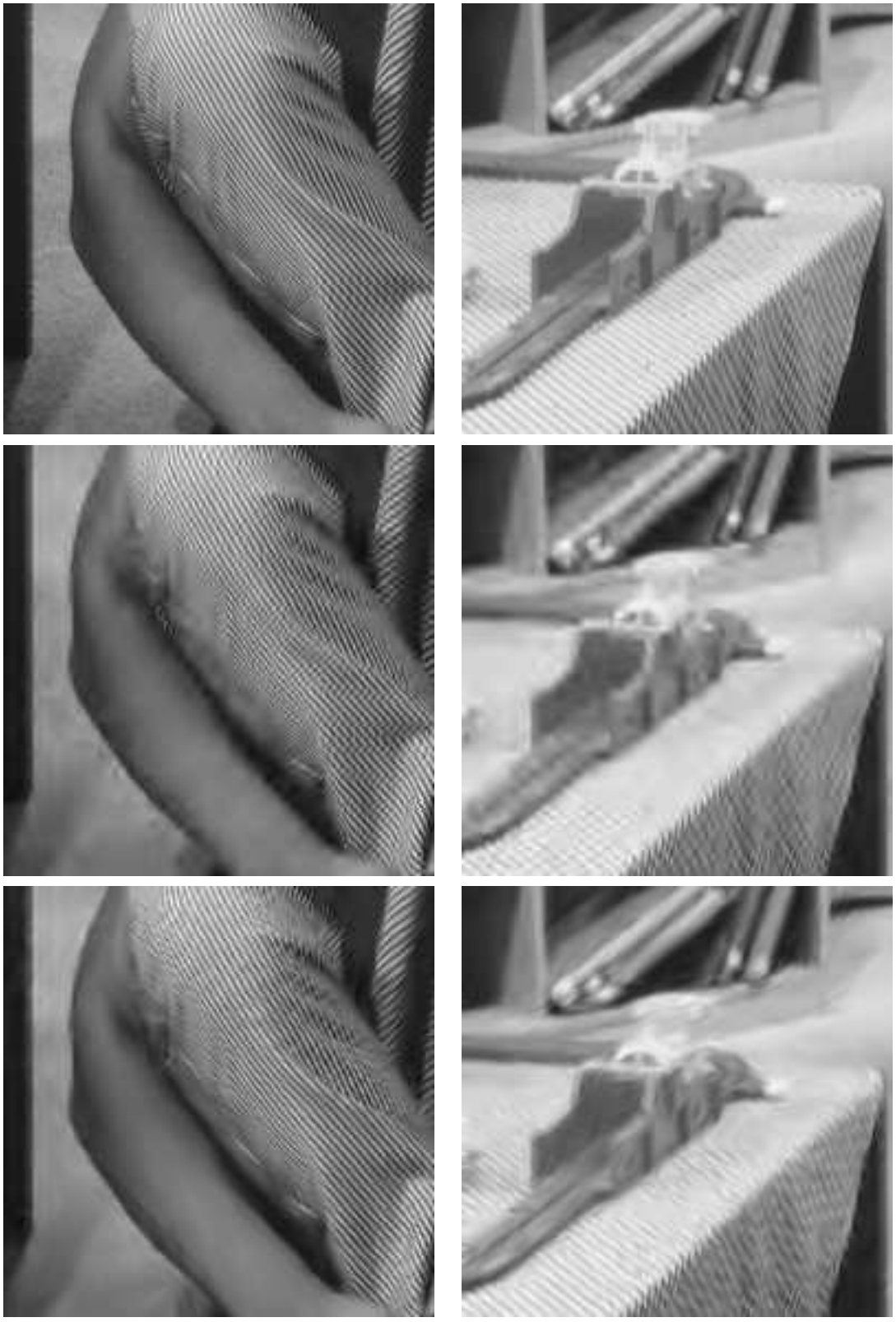

FIG. 4.10: Détails de l'image *barbara* originale [haut], analysée par ondelettes séparables [milieu] et par ondelettes orientées [bas], puis codée à 0.3 bpp par le codeur EBCOT. Les contours sont mieux préservés par les ondelettes orientées et l'effet de rebond est limité.

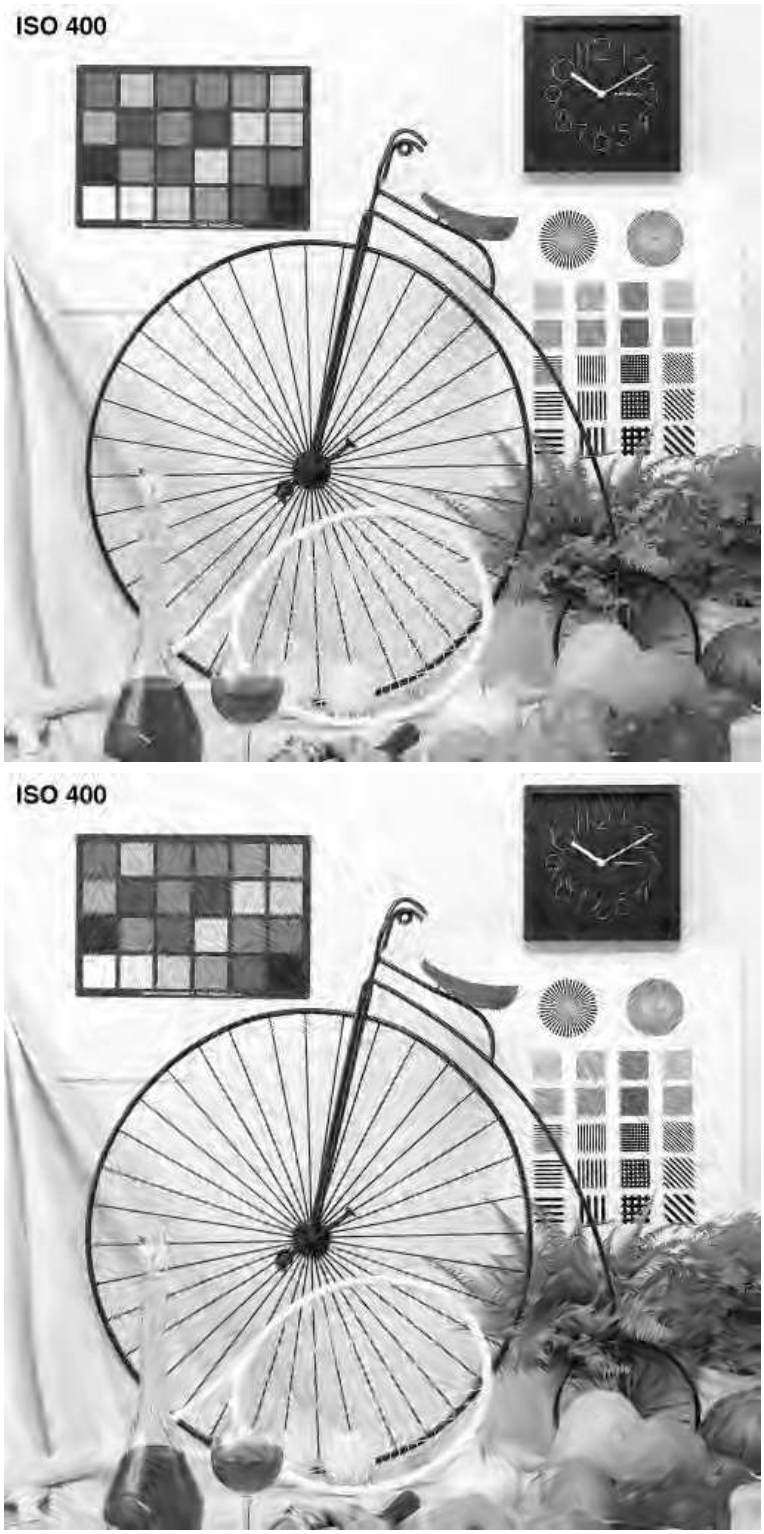

FIG. 4.11: [haut] Image *bike* 512x512 codée en utilisant 5 niveaux de décomposition d'ondelettes 9/7 séparables et le codeur EBCOT à 0.3 bpp, PSNR = 25.14 dB. [bas] Même image codée en utilisant 10 niveaux d'ondelettes 9/7 orientées et le codeur EBCOT à 0.3 bpp, PSNR = 25.00 dB.

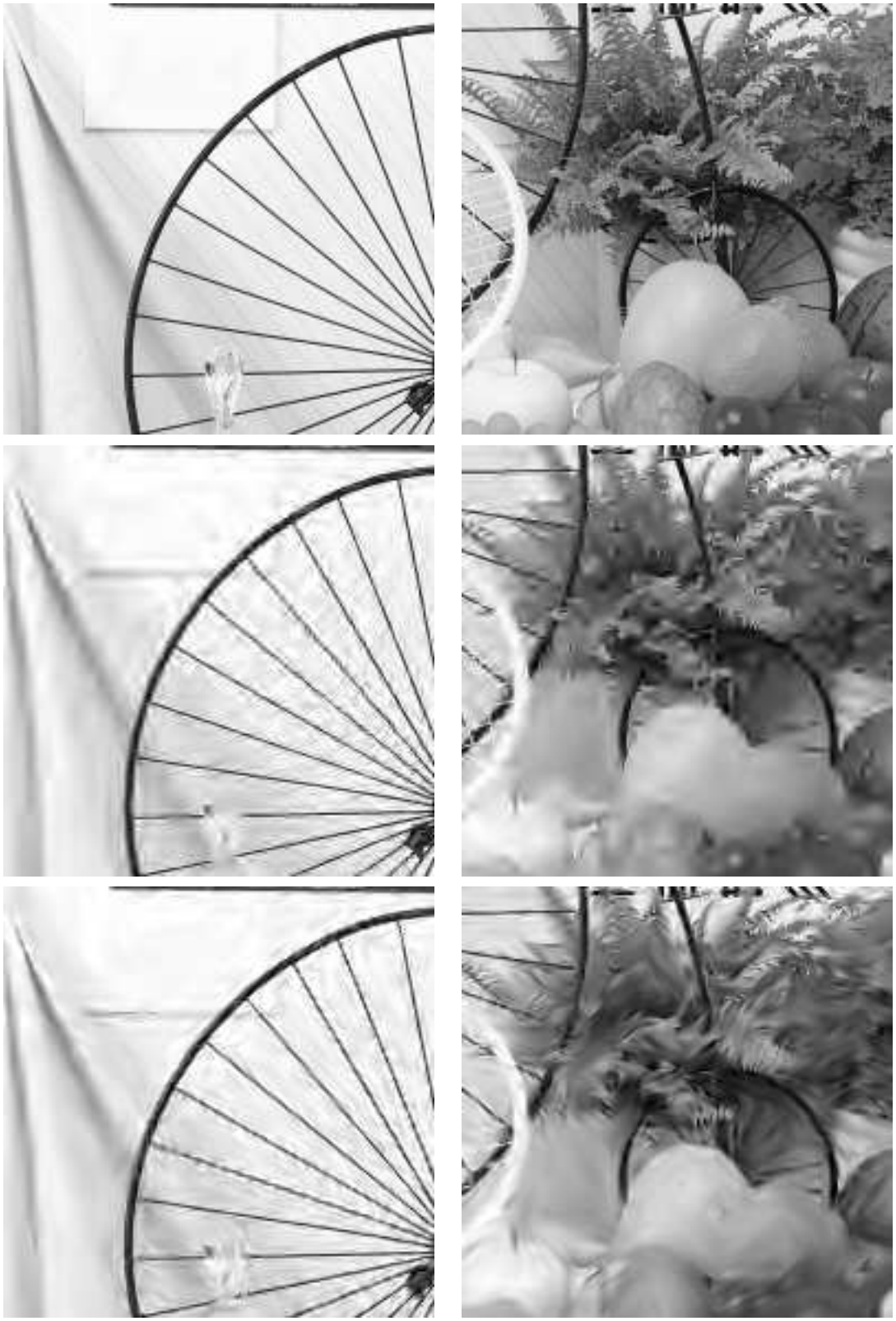

FIG. 4.12: Détails de l'image *bike* originale [haut], analysée par ondelettes séparables [milieu] et par ondelettes orientées [bas], puis codée à 0.3 bpp par le codeur EBCOT. Les textures apparaissent moins pixélisées avec les ondelettes orientées.

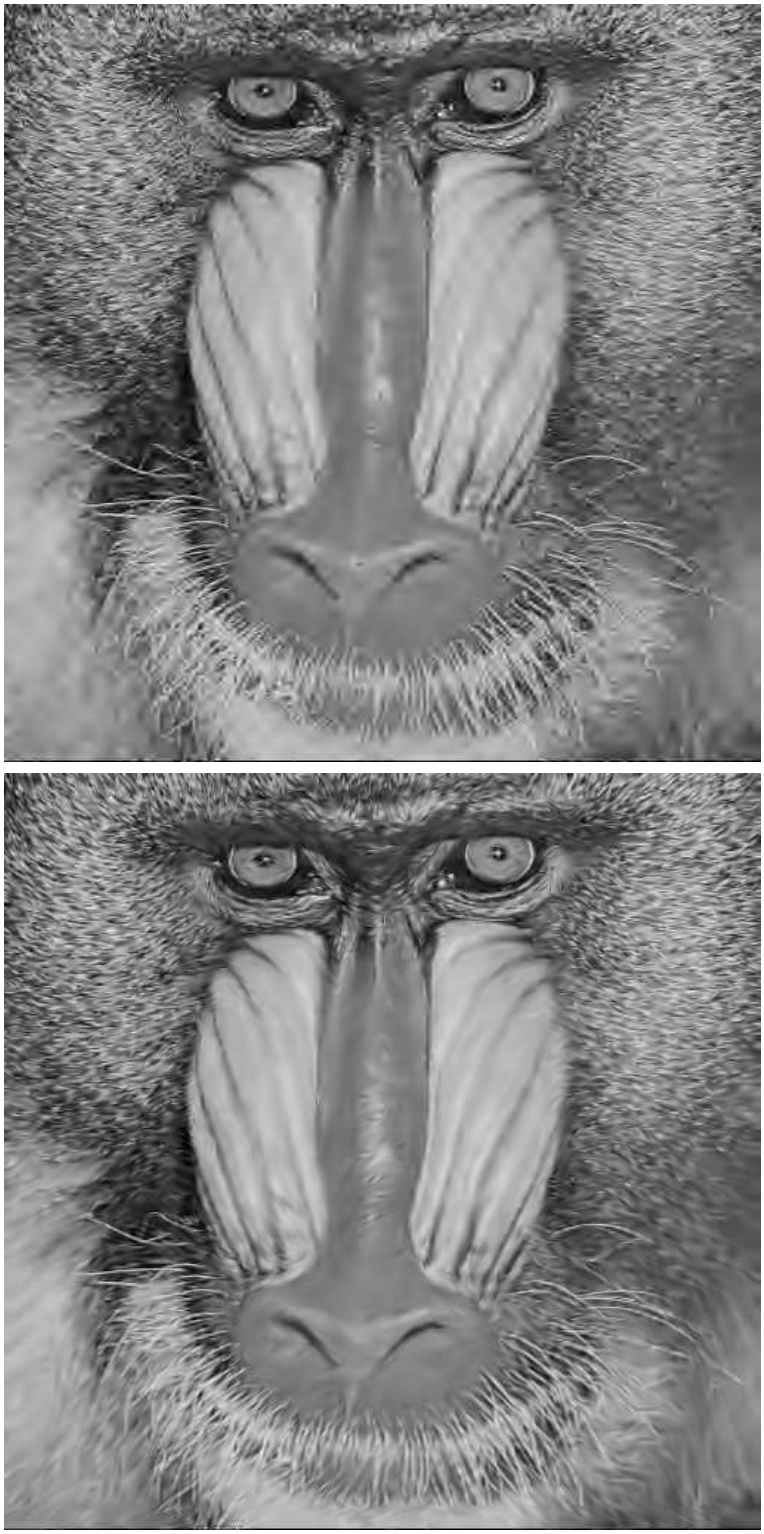

FIG. 4.13: [haut] Image *mandrill* 512x512 codée en utilisant 5 niveaux de décomposition d'ondelettes 9/7 séparables et le codeur EBCOT à 0.3 bpp, PSNR = 23.64 dB. [bas] Même image codée en utilisant 10 niveaux d'ondelettes 9/7 orientées et le codeur EBCOT à 0.3 bpp, PSNR = 23.88 dB.

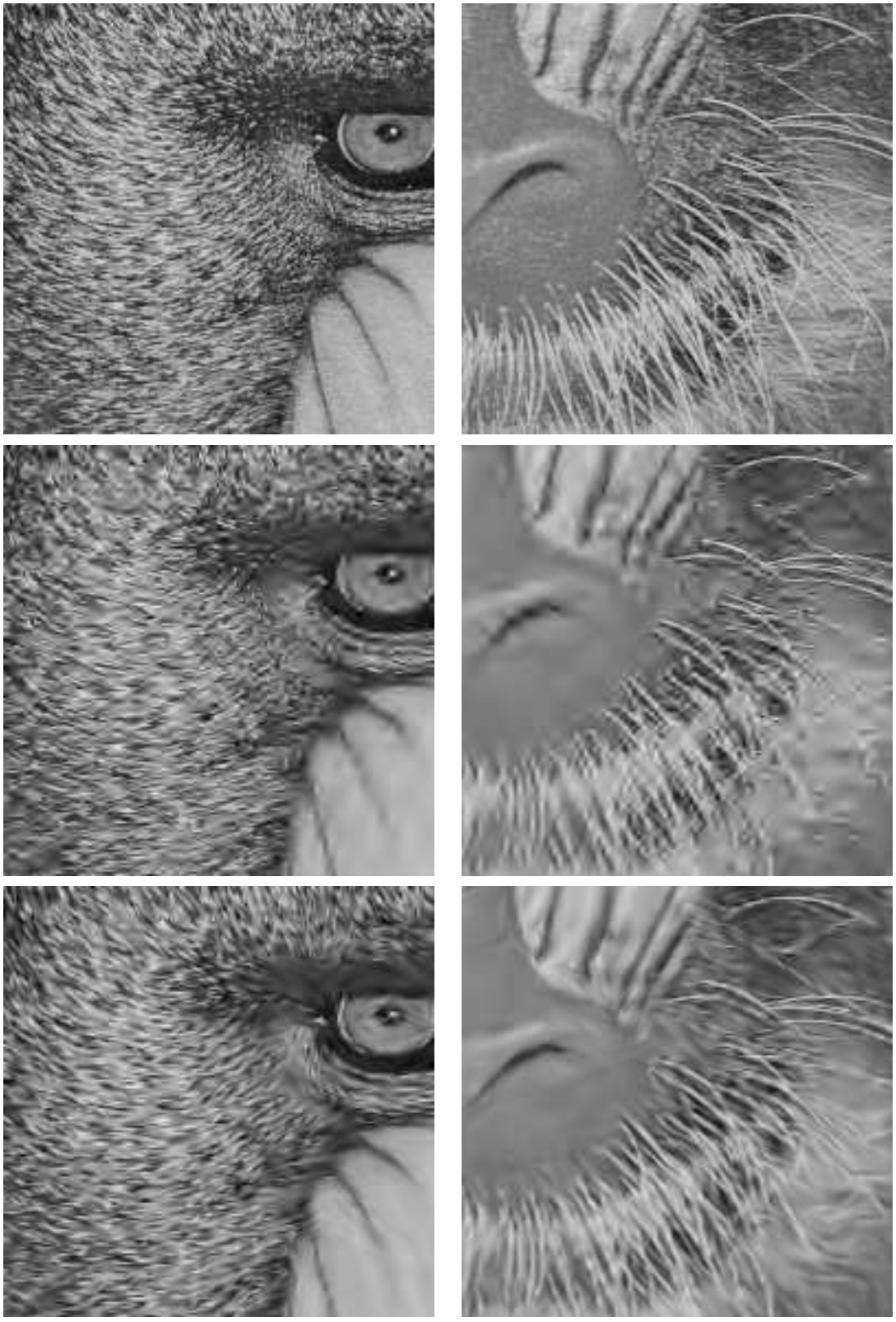

FIG. 4.14: Détails de l'image *mandrill* originale [haut], analysée par ondelettes séparables [milieu] et par ondelettes orientées [bas], puis codée à 0.3 bpp par le codeur EBCOT. Les structures orientées sont préservées à toutes échelles par les ondelettes orientées. Les textures conservent également leur orientation générale.

FIG. 4.15: [haut] Image *goldhill* 512x512 codée en utilisant 5 niveaux de décomposition d'ondelettes 9/7 séparables et le codeur EBCOT à 0.3 bpp, PSNR = 31.18 dB. [bas] Même image codée en utilisant 10 niveaux d'ondelettes 9/7 orientées et le codeur EBCOT à 0.3 bpp, PSNR = 30.88 dB.

FIG. 4.16: Détails de l'image *goldhill* originale [haut], analysée par ondelettes séparables [milieu] et par ondelettes orientées [bas], puis codée à 0.3 bpp par le codeur EBCOT. Bien que les structures fines soient bien préservées par les ondelettes orientées, le bruit dans les zones uniformes explique probablement la perte observée en PSNR.

FIG. 4.17: Codage de l'image *lena* par EBCOT après décomposition en ondelettes séparables [gauche] et orientée [droite]. Les débits et les PSNR pour la transformée en ondelettes séparables et orientées sont respectivement de [haut] 33.33 dB et 33.64 dB à 0.2 bpp, [milieu] 37.55 dB et 37.58 dB à 0.5 bpp, et [bas] 40.51 dB et 40.42 dB à 1.0 bpp.

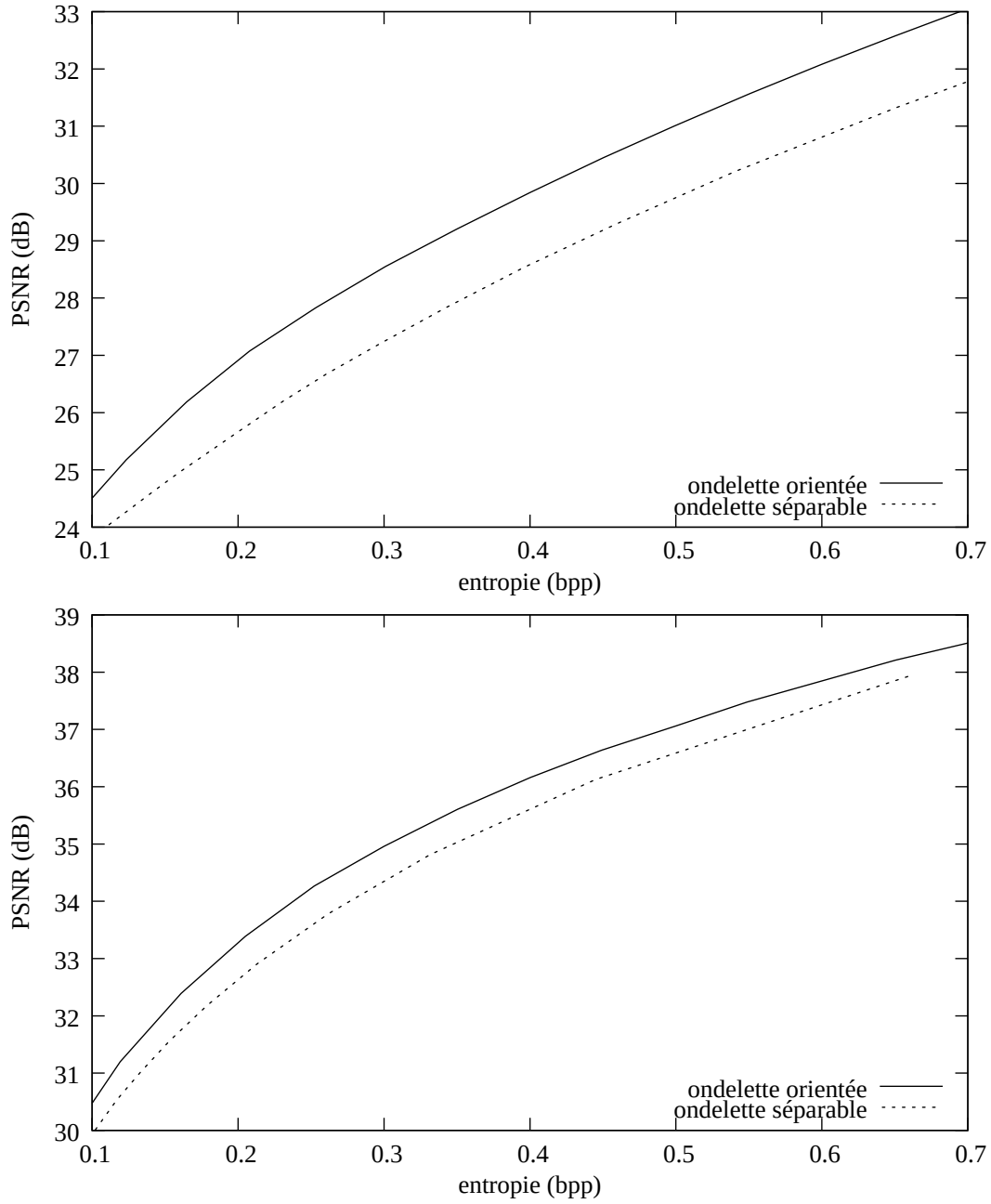

FIG. 4.18: Performance débit-distorsion en utilisant l'estimateur de débit par entropie absolue pour les images *barbara* [haut] et *lena* [bas] codées avec 5 niveaux d'ondelettes séparables et 10 niveaux d'ondelettes orientées respectivement.

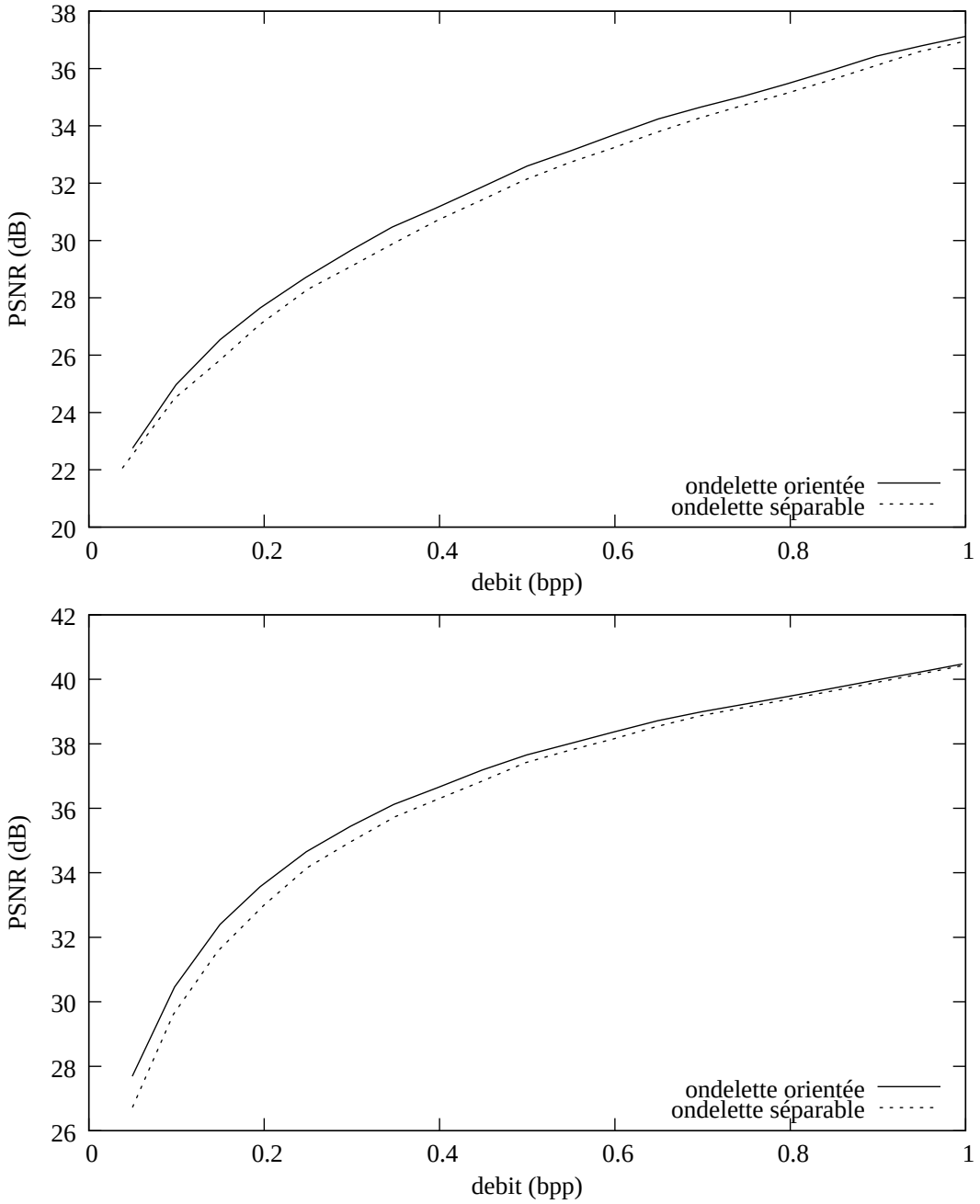

FIG. 4.19: Performance débit-distorsion en utilisant EBCOT pour les images *barbara* [haut] et *lena* [bas] codées avec 5 niveaux d'ondelettes séparables et 10 niveaux d'ondelettes orientées respectivement.

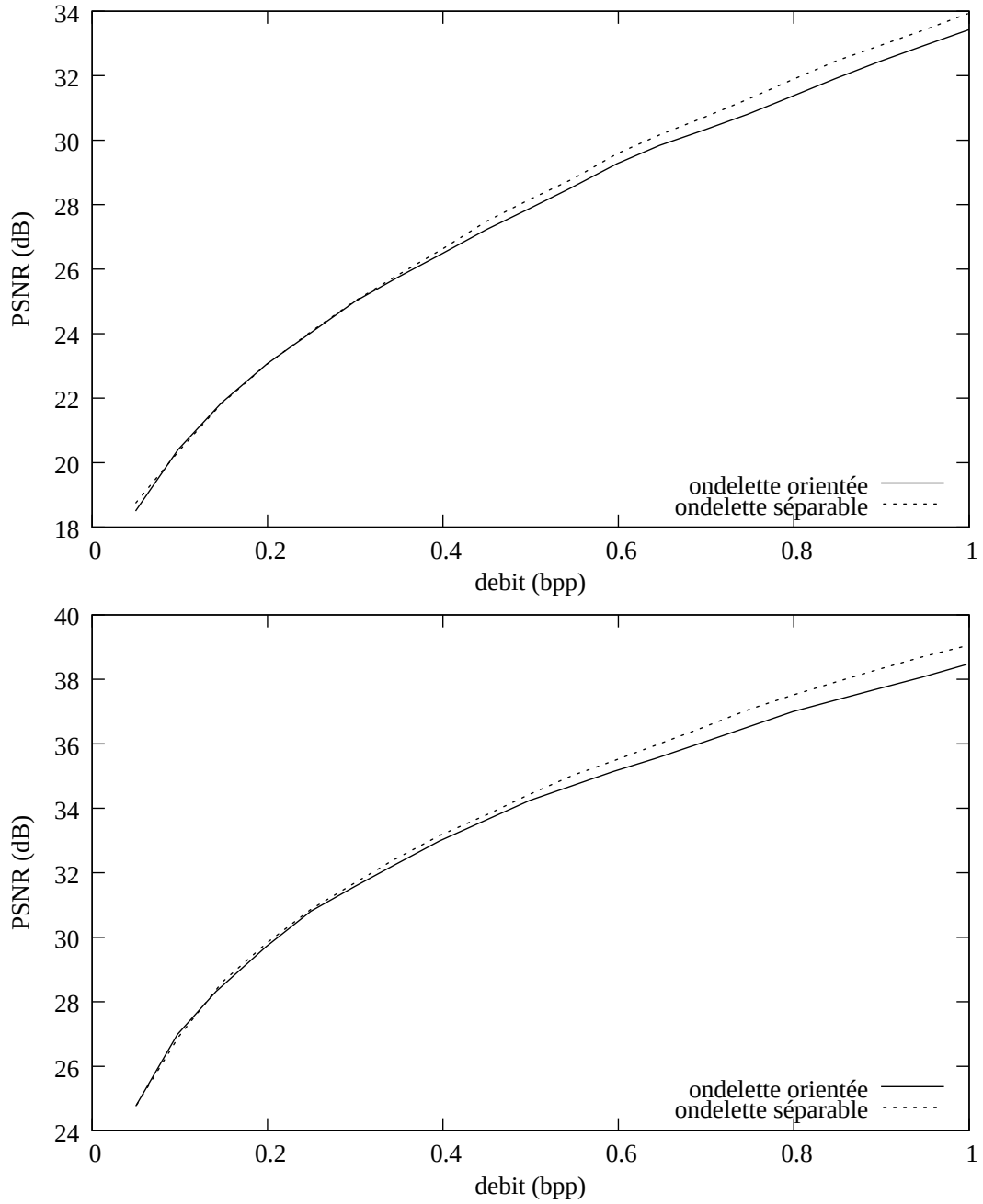

FIG. 4.20: Performance débit-distorsion en utilisant EBCOT pour les images *bike* [haut] et *boat* [bas] codées avec 5 niveaux d'ondelettes séparables et 10 niveaux d'ondelettes orientées respectivement.

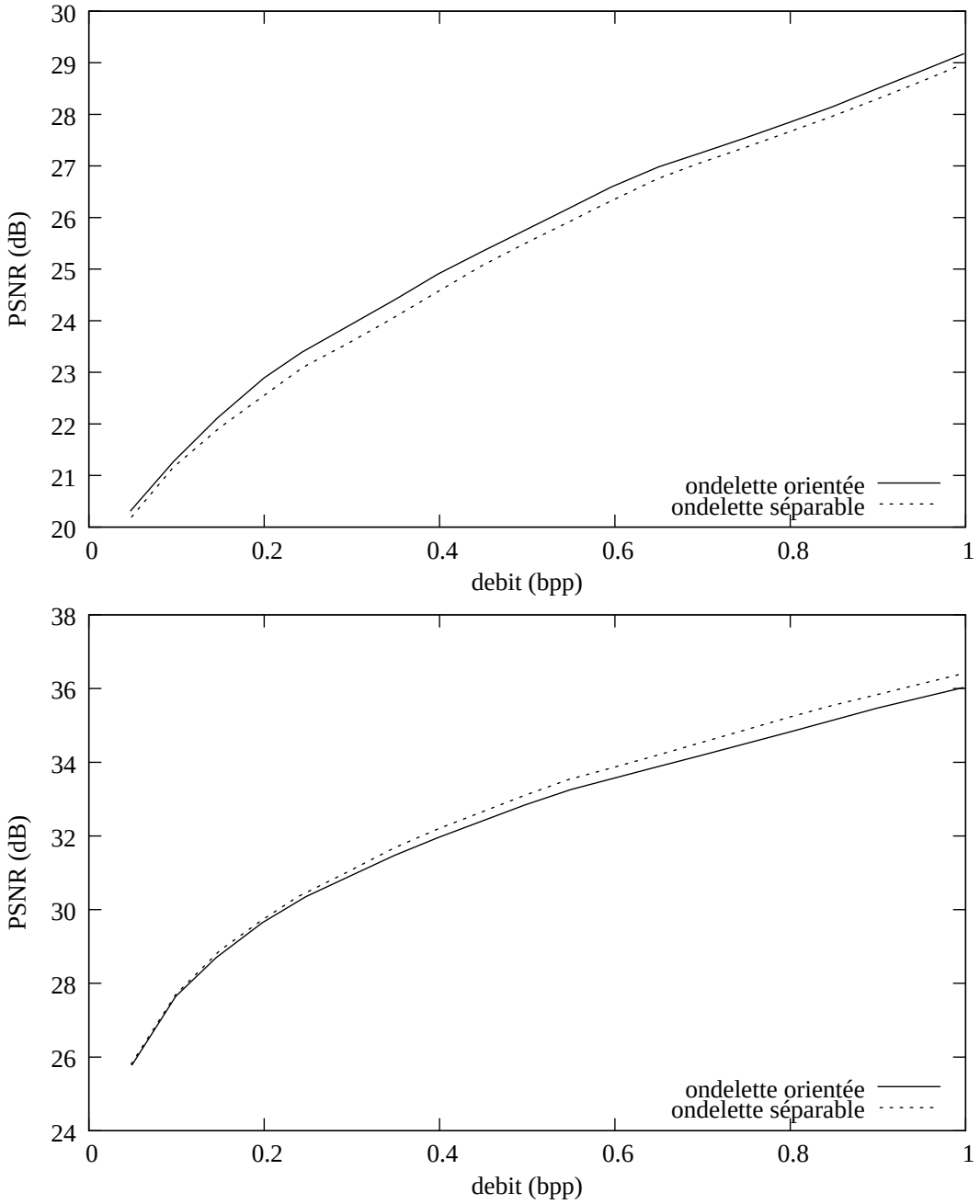

FIG. 4.21: Performance débit-distorsion en utilisant EBCOT pour les images *mandrill* [haut] et *goldhill* [bas] codées avec 5 niveaux d'ondelettes séparables et 10 niveaux d'ondelettes orientées respectivement.

4.2.6 Étude de la dépendance résiduelle

Les résultats en compression avec le codeur EBCOT suggèrent que les coefficients d'ondelettes orientées sont moins dépendants que les coefficients d'ondelettes séparables. Ainsi, des modèles contextuels d'ordre faible devraient être suffisants pour exploiter les dépendances résiduelles entre les coefficients. De plus, l'hypothèse que la dépendance intra échelle est suffisante pour exploiter la plupart de la dépendance résiduelle des coefficients d'ondelettes [105] pourrait être invalidée dans le cas des ondelettes orientées.

Par conséquent, en suivant une procédure similaire à celle exposée dans [105], nous utilisons l'information mutuelle entre les coefficients d'ondelettes orientées comme mesure de leur dépendance résiduelle. Nous rappelons que cette quantité correspond ici au surcoût de codage induit par l'hypothèse d'indépendance des coefficients.

Pour chaque coefficient X, nous considérons son coefficient parent $\mathcal{P}(X)$ et ses 8 plus proches voisins intra sous-bande $\mathcal{N}(X)$. La loi jointe $\mathbb{P}(X, \mathcal{P}(X), \mathcal{N}(X))$ est estimée à partir d'histogrammes sur les sous-bandes. Cette loi varie entre les différentes sous-bandes, mais est supposée stationnaire au sein d'une même sous-bande. Du fait du nombre limité de coefficients disponibles dans chaque sous-bande et de la dimension élevée de la loi jointe, le voisinage $\mathcal{N}(X)$ est tout d'abord projeté dans $\mathbb{R}$ en utilisant une fonction de projection f pour s'assurer de la pertinence statistique des histogrammes. La loi tridimensionnelle $\mathbb{P}(X, \mathcal{P}(X), f(\mathcal{N}(X)))$ est alors estimée au lieu de $\mathbb{P}(X, \mathcal{P}(X), \mathcal{N}(X))$. D'après le théorème de traitement des données,

$$I(X; \mathcal{P}(X), f(\mathcal{N}(X))) \leq I(X; \mathcal{P}(X), \mathcal{N}(X)),$$

où la perte d'information dépend du choix de la fonction de projection f. Puisque l'information mutuelle est invariante aux traitements inversibles, les histogrammes sont estimées dans le domaine logarithmique sur $\tilde{X} = \log(|X|)$, $\tilde{\mathcal{P}}(X) = \log(|\mathcal{P}(X)|)$ et $\tilde{\mathcal{N}}(X) = \log(|f(\mathcal{N}(X))|)$ en utilisant 8 cases par variable. On obtient donc un espace de $8^3 = 512$ cases pour $256 \times 256 = 65536$ coefficients dans la sous-bande de plus haute fréquence des images 512x512 de test.

L'information mutuelle entre X et ses voisins $\mathcal{P}(X)$, $\mathcal{N}(X)$ est donc finalement estimée par

$$\hat{I}(X; \mathcal{P}(X), \mathcal{N}(X)) \approxeq \sum \hat{\mathbb{P}}(\tilde{X}, \tilde{\mathcal{P}}(X), \tilde{\mathcal{N}}(X)) \log \left(\frac{\hat{\mathbb{P}}(\tilde{X}, \tilde{\mathcal{P}}(X), \tilde{\mathcal{N}}(X))}{\hat{\mathbb{P}}(\tilde{X}) \hat{\mathbb{P}}(\tilde{\mathcal{P}}(X), \tilde{\mathcal{N}}(X))} \right),$$

où la somme s'effectue sur l'ensemble des cases de l'histogramme et où $\hat{\mathbb{P}}(X)$ et $\hat{\mathbb{P}}(\tilde{\mathcal{P}}(X), \tilde{\mathcal{N}}(X))$ sont obtenues à partir de la loi jointe par intégration sur $(\tilde{\mathcal{P}}(X), \tilde{\mathcal{N}}(X))$ et X respectivement. Les estimées d'information mutuelle $\hat{I}(X; \mathcal{P}(X))$ et $\hat{I}(X; \mathcal{N}(X))$ sont également calculées pour déterminer la proportion d'information contenue dans le coefficient parent $\mathcal{P}(X)$ et les coefficients voisins $\mathcal{N}(X)$.

Il a été montré dans [105] qu'un estimateur local de la variance fournit une bonne prédiction statistique de l'amplitude du coefficient. Par conséquent, nous avons choisi

par la suite d'utiliser la fonction de projection suivante :

$$f(\mathcal{N}(X)) = \sum_{N \in \mathcal{N}(X)} N^2,$$

où $\mathcal{N}_i(X)$ représente un coefficient du voisinage intra échelle. Dans les codeurs de sous-bandes actuels, la fonction f est souvent moins informative et basée sur le concept de signifiance, qui correspond à un seuillage des coefficients voisins avant projection sur un faible nombre de contextes.

Le tableau 4.3 compare la quantité d'information mutuelle présente entre les coefficients des sous-bandes de haute fréquence d'ondelettes 9/7 séparables et orientées. L'information résiduelle moyenne dans les sous-bandes HL,LH,HH et H0H,H0L,H1H,H1L est indiquée pour la transformée en ondelettes séparables et la transformée en ondelettes orientées respectivement. L'information mutuelle entre les coefficients d'ondelettes orientées est 20 à 50 % moindre comparée à l'information mutuelle présente entre les coefficients d'ondelettes séparables, et montre que la dépendance résiduelle exploitable par le codeur EBCOT est plus limitée. Pour les deux transformée, la dépendance entre coefficients intra-échelle est plus importante que la dépendance entre coefficients inter-échelle. Ceci suggère que des codeur par blocs, comme EBCOT, sont également mieux adaptés au codage des coefficients d'ondelettes orientées que les codeurs par arbres, comme EZW, bien que le gain attendu comparé à l'utilisation de codeurs plus simples soit moindre que dans le cas des ondelettes séparables.

Le tableau 4.3 montre également l'impact de la taille du voisinage intra échelle sur l'information mutuelle. Les différents ensembles de voisinages intra considérés sont présentés dans la figure 4.22. La décroissance de l'information mutuelle due à la réduction du voisinage intra suit un comportement similaire dans toutes les sous-bandes que ce soit pour les ondelettes orientées ou pour les ondelettes séparables. L'impact sur l'information mutuelle de l'utilisation du voisinage $\mathcal{N}_4$ restreint à 4 voisins au lieu du voisinage complet $\mathcal{N}_8$ est négligeable. Ceci suggère qu'un tel voisinage est suffisant pour extraire la dépendance résiduelle. On constate également que le voisinage causal $\mathcal{N}_{c4}$ est capable d'extraire plus d'information résiduelle dans le cas des ondelettes orientées que dans le cas des ondelettes séparables, proportionnellement à l'information résiduelle totale présente

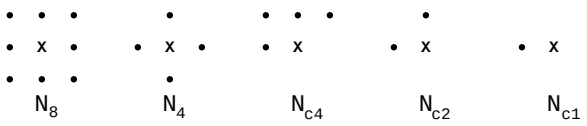

FIG. 4.22: Ensemble de coefficients intra échelle. Le coefficient considéré est représenté par une croix.

image	lena		barbara	
transformée	séparable (bpp)	orientée (bpp)	séparable (bpp)	orientée (bpp)
$\hat{I}(X;\mathcal{N}_8,P)$	0.195	0.162	0.506	0.333
$\hat{I}(X;P)$	0.102	0.080	0.147	0.129
$\hat{I}(X;\mathcal{N}_8)$	0.181	0.146	0.493	0.318
$\hat{I}(X;\mathcal{N}_4)$	0.170	0.127	0.491	0.307
$\hat{I}(X;\mathcal{N}_{c4})$	0.146	0.118	0.441	0.277
$\hat{I}(X;\mathcal{N}_{c2})$	0.123	0.091	0.403	0.243
$\hat{I}(X;\mathcal{N}_{c1})$	0.074	0.052	0.300	0.162

TAB. 4.3: Information mutuelle moyenne entre les coefficients d'ondelettes des sous-bandes de plus hautes fréquences pour différents voisinages.

4.3 Application au débruitage d'image

4.3.1 Modèle markovien

Dans le contexte du débruitage, la carte d'orientation est modélisée par un champ de Markov caché (HMF) sur $(L^k_{\mathcal{H}})$. Chaque noeud du champ a une orientation binaire à la position n. À une échelle donnée k, un noeud possède 8 voisins au sein de cette même échelle, modélisant les dépendances intra échelle entre les orientations. De plus, un noeud est lié a son noeud père à l'échelle $k-2$ et ses quatre noeuds fils à l'échelle $k+2$ quand ils existent. Ceci permet de modéliser les dépendances inter échelle entre les orientations. Par conséquent, chaque noeud a 13 voisins, auxquels correspondent 41 cliques. Les fonctions de potentiel sont définies de manière identique pour toutes les cliques $\mathcal{C}(\mathcal{N}_{\mathsf{n}})$ du voisinage $\mathcal{N}_{\mathsf{n}}$ du noeud n :

$$\forall c \in \mathcal{C}(\mathcal{N}_i), \phi(m_c) = \begin{cases} -\alpha & \text{si } \forall m_{\mathsf{j}} \in m_{c\setminus\{\mathsf{i}\}}, m_{\mathsf{j}} = m_{\mathsf{i}} \\ +\alpha & \text{sinon} \end{cases}.$$

Ce modèle de Markov correspond à l'approximation suivante (Fig. 4.23) :

$$\forall \mathsf{n} \in \mathbb{Z}^2, \mathbb{P}(M_{\mathsf{n}}|M_{\mathbb{Z}^2\setminus\{\mathsf{n}\}}) = \mathbb{P}(M_{\mathsf{n}}|M_{\mathcal{N}_{\mathsf{n}}\setminus\{\mathsf{n}\}}).$$

Soit $\mathcal{N}^\star_{\mathsf{n}} = \mathcal{N}_{\mathsf{n}}\setminus\{\mathsf{n}\}$. Le théorème d'Hammersley-Clifford [106] donne la loi conditionnelle de M_{n} sachant ses voisins $M_{\mathcal{N}^\star_{\mathsf{n}}}$ à une constante inconnue γ près :

$$\mathbb{P}(M_{\mathsf{n}}|M_{\mathcal{N}^\star_{\mathsf{n}}}) = \gamma\, e^{-\sum\limits_{c\in\mathcal{C}(\mathcal{N}_{\mathsf{n}})}\phi(m_c)}. \tag{4.4}$$

Pour une observation Y des coefficients d'ondelettes, la carte d'orientation est obtenue en maximisant la vraisemblance $\mathbb{P}(M|Y)$ parmis toutes les cartes M possibles. Ceci s'effectue en minimisant l'énergie de Gibbs associée au champ de Markov par un échantillonage de Gibbs. En supposant que les observations (coefficients d'ondelettes) sont

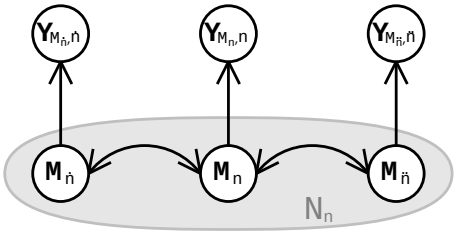

FIG. 4.23: Graphe de dépendance entre les variables aléatoires. Le champ de Markov est représenté comme une chaîne ici pour des raisons de clarté. L'observation $Y_{\mathsf{n},M_\mathsf{n}}$ est indépendante des autres réalisations conditionnellement à M_n. La dépendance entre les états cachés M_n est limitée au voisinage $\mathcal{N}_\mathsf{n}$.

indépendantes sachant la carte M (Fig. 4.23) et qu'elles suivent une distribution gaussienne, nous avons :

$$\mathbb{P}(M_\mathsf{n}, Y_{\mathsf{n},m_\mathsf{n}}|M_{\mathcal{N}_\mathsf{n}^\star}) = \mathbb{P}(M_\mathsf{n}|M_{\mathcal{N}_\mathsf{n}^\star})\mathbb{P}(Y_{\mathsf{n},m_\mathsf{n}}) \tag{4.5}$$

$$= \gamma\, e^{-\sum\limits_{c\in\mathcal{C}(\mathcal{N}_\mathsf{n})}\phi(m_c) - \frac{\|y_{\mathsf{n},m_\mathsf{n}}\|^2}{2\sigma_Y^2}}, \tag{4.6}$$

où σ_Y^2 est la variance des coefficients d'ondelettes. Pour l'application de débruitage, σ_Y^2 est estimée à partir de la variance empirique, indépendamment pour chaque sous-bande. Dans l'équation 4.6, $\sum\limits_{c\in\mathcal{C}(\mathcal{N}_\mathsf{n})}\phi(m_c)$ correspond au terme de régularisation, tandis que $\frac{\|y_{\mathsf{n},m_\mathsf{n}}\|^2}{2\sigma_Y^2}$ correspond au terme d'attache aux données.

La carte m est tout d'abord initialisée avec les orientations minimisant l'énergie des coefficients d'ondelettes (correspondant à $\alpha = 0$). Elle est ensuite ré-échantillonnée de manière itérative en mettant à jour chaque noeud de la carte, dans un ordre aléatoire. Pour chaque noeud n, la probabilité suivante est calculée pour toutes les orientations :

$$\mathbb{P}(M_\mathsf{n}|M_{\mathcal{N}_\mathsf{n}^\star}, Y_{\mathsf{n},m_\mathsf{n}}) = \frac{\mathbb{P}(M_\mathsf{n}, Y_{\mathsf{n},m_\mathsf{n}}|M_{\mathcal{N}_\mathsf{n}^\star})}{\mathbb{P}(Y_{\mathsf{n},m_\mathsf{n}}|M_{\mathcal{N}_\mathsf{n}^\star})}$$
$$= \frac{\mathbb{P}(M_\mathsf{n}, Y_{\mathsf{n},m_\mathsf{n}}|M_{\mathcal{N}_\mathsf{n}^\star})}{\mathbb{P}(M_\mathsf{n}=0, Y_{\mathsf{n},m_\mathsf{n}}|M_{\mathcal{N}_\mathsf{n}^\star}) + \mathbb{P}(M_\mathsf{n}=1, Y_{\mathsf{n},m_\mathsf{n}}|M_{\mathcal{N}_\mathsf{n}^\star})}.$$

Un nouvelle orientation m_n est tirée selon cette distribution. Après quelques itérations, le champ se stabilise en une réalisation de M sachant Y. Une nouvelle décomposition en ondelettes est calculée en utilisant la nouvelle carte, donnant une réalisation de Y sachant M. Ce processus global est itéré quelques fois avant l'étape finale de débruitage.

Afin de réaliser ces calculs, le paramètre de régularisation α doit être défini. Bien qu'il puisse être estimé en utilisant un algorithme d'Expectation-Maximization (EM), pour des raisons de simplicité nous choisissons ici de le fixer à la valeur $\alpha = 0.3$. Nous avons observé que ce paramètre est stable quelque soit l'image et que le choix d'une valeur particulière n'impacte pas significativement les résultats expérimentaux.

4.3.2 Débruitage

Nous avons réalisé des expériences de débruitage simples pour comparer la transformée en ondelettes orientées aux autres transformées comme les transformées en contourlettes, en bandelettes ou en ondelettes séparables. Nous comparons aussi notre méthode à d'autres techniques de débruitage récentes [107] [108]. Dans [107] (LCHMM-SI), les coefficients d'ondelettes séparables sont supposés suivre un modèle de mélange de gaussiennes. Un champs de Markov est utilisé pour classifier les coefficients en deux classes de différente variance et le débruitage est effectué en utilisant un algorithme EM relativement complexe. Dans [108] (SP-GSM), une pyramide orientable, hautement redondante, est utilisée pour la décomposition en sous-bandes, et combinée avec un modèle de mélange de gaussiennes pour le débruitage.

Dans toutes les expériences, un bruit blanc gaussien de variance σ_N^2 est ajouté à l'image. Contrairement à [107] et [108], le débruitage est effectué ici au moyen d'un simple seuillage des coefficients d'ondelettes, à savoir :

$$\hat{y}_{\mathsf{n},m_\mathsf{n}} = \begin{cases} 0 & \text{si } |y_{\mathsf{n},m_\mathsf{n}}| < \beta\sigma_N \\ y_{\mathsf{n},m_\mathsf{n}} & \text{sinon} \end{cases} ,$$

où β est un facteur constant fixé à $\beta = 3$ pour toutes les expériences.

Afin de s'affranchir du problème de non invariance par translation des transformées à échantillonage critique (toutes exceptée la transformée en contourlettes), nous appliquons notre méthode de débruitage sur 16 versions translatées de l'image bruitée (+0 à 3 pixels dans les directions verticales et horizontales). Les images débruitées sont ensuite translatées à leur position initiale puis moyennées pour obtenir l'image débruitée finale. Ces conditions sont utilisées pour obtenir les résultats expérimentaux pour les ondelettes orientées, les contourlettes et les ondelettes séparables, tandis que les résultats des bandelettes, et des techniques LCHMM-SI et SP-GSM proviennent des publications [84] [109] [107] [108]. Le tableau 4.4 montre qu'on obtient des performances meilleures ou identiques aux autres techniques par transformées en utilisant la transformée en ondelettes orientées. De plus, ces performances sont proches des meilleures techniques de l'état de l'art en débruitage. La figure 4.25 montre les résultats de la méthode de débruitage appliquée à l'image *lena*. Le rapport signal à bruit est amélioré de 10.4 dB pour un bruit d'écart type 25. La figure 4.26 compare cette même technique au débruitage par ondelettes séparables. Un gain de 0.4 dB est observé et les contours de l'image reconstruite sont visuellement mieux restitués.

image	lena			
σ_N	15	20	25	40
bruitée (dB)	24.6	22.1	20.2	16.4
ondelette (dB)	32.5	31.0	29.8	27.4
contourlette (dB)	32.3	30.9	29.8	27.4
bandelette [84] [109] (dB)	-	-	30.3	27.0
orientée (dB)	33.1	31.7	30.6	28.1
LCHMM-SI [107] (dB)	33.0	31.7	30.6	-
SP-GSM [108] (dB)	33.9	32.6	31.7	-

image	barbara			
σ_N	15	20	25	40
bruitée (dB)	24.6	22.2	20.3	16.5
ondelette (dB)	29.6	27.9	26.6	24.1
contourlette (dB)	30.3	28.8	27.6	25.1
orientée (dB)	30.6	29.0	27.8	25.2
LCHMM-SI [107] (dB)	31.4	29.7	28.5	-
SP-GSM [108] (dB)	31.9	30.3	29.1	-

TAB. 4.4: Comparaison des différentes transformées pour le débruitage des images *lena* et *barbara*. Le PSNR des images reconstruites est donné.

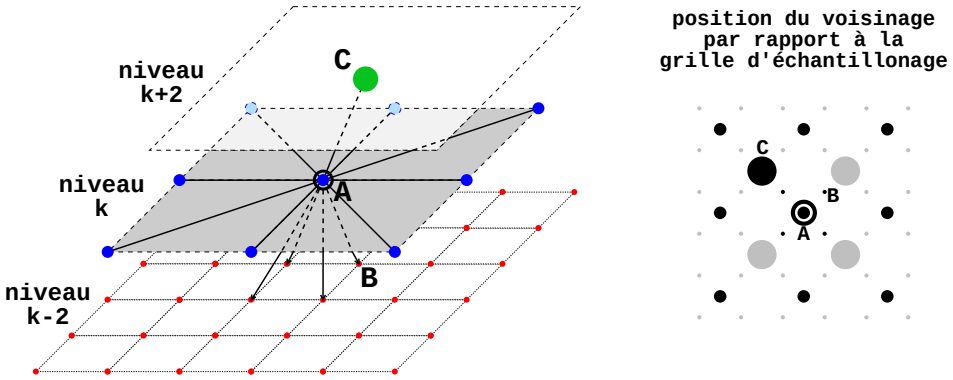

FIG. 4.24: Voisinage considéré dans le champ de Markov. La figure 3D représente l'ensemble des voisins du noeud central (A) au niveau k en lignes épaisses. La figure 2D indique en noir la position spatiale de ce même voisinage dans $\mathbb{Z}^2$. Les points les plus petits correspondent aux échelles fines. Le noeud (B) est fils de (A), tandis que le noeud (C) est père de (A).

4.4 Conclusion

Nous avons présenté une nouvelle transformée adaptative discrète basée sur le schéma lifting appliqué le long d'une carte d'orientation définie sur une grille multirésolution quinconce. Les performances de cette nouvelle transformée ont été évaluées dans le contexte de la compression et du débruitage. En fonction de l'application, la carte est soit codée en utilisant un quad-tree optimisé selon un critère débit-distorsion, soit estimée en utilisant un champ de Markov. La transformée en ondelettes orientées offre des performances similaires ou meilleures que les autres transformées, pour une complexité réduite.

L'information mutuelle entre les coefficients d'ondelettes orientées a été mesurée. Ceci devrait permettre la définition de modèles contextuels appropriés pour le codage de ces coefficients en utilisant les codeurs de sous-bande actuels. Dans le contexte du débruitage, les performances pourraient être encore améliorées en utilisant des modèles non gaussiens à la fois pour la recherche de la carte d'orientation dans l'image bruitée et pour l'étape de suppression du bruit.

Le nombre d'orientations et la structure d'échantillonnage sont fixés dans la transformée proposée. Une plus grande souplesse en termes d'orientations et de grilles d'échantillonnages pourrait en améliorer les performances, en s'attachant également à conserver l'orthogonalité. Un choix de différents filtres pour chaque échelle pourrait de plus améliorer la concentration d'énergie dans les basses fréquences. Il serait également intéressant d'ajouter aux choix d'orientations la possibilité d'utiliser une ondelette quinconce non orientée pour améliorer la qualité des images dans les zones uniformes. Toutefois, le coût de codage supplémentaire de ce mode dans les cartes d'orientations pourrait compenser le gain de codage obtenu sur les coefficients d'ondelettes.

FIG. 4.25: Expérience de débruitage. [haut] L'image *lena* bruitée par un bruit blanc additif gaussien d'écart type $\sigma_N = 25$ (PSNR = 20.2dB). [bas] L'image reconstruite après débruitage par ondelettes orientées (PSNR = 30.6dB).

FIG. 4.26: Débruitage de l'image *café*. [haut gauche] Image originale, [haut droit] image bruitée par un bruit blanc additif gaussien d'écart type $\sigma_N = 40$ (PSNR = 16.75 dB), [bas gauche] image reconstruite après débruitage par ondelettes séparables (PSNR = 21.03 dB), [bas droit] image reconstruite après débruitage par ondelettes orientées (PSNR = 21.47 dB).

Chapitre 5

Turbo TCQ

La quantification codée par treillis (TCQ) est une technique de quantification vectorielle rapide reposant sur un code convolutif. Sous une forme adaptée au codage des sous-bandes d'ondelettes [110] [111] [112] [113], elle améliore les performances à haut débit de manière significative par rapport à la quantification scalaire par zone morte. Elle donne également une quantification emboîtée en la combinant à un codage par plans de bits suivi d'un codeur arithmétique contextuel ou d'un codeur par zerotree comme SPIHT [49]. Cette technique a par ailleurs été adoptée récemment en tant qu'extension de la norme JPEG2000.

Dans ce chapitre nous étudions l'application du principe turbo à la TCQ. En s'appuyant sur la notion de partitions d'ensembles proposée par Ungerboeck en modulation codée par treillis (TCM), nous utilisons un turbo code pour le partitionement d'un dictionnaire de quantification de manière à obtenir le débit désiré. La structure du quantificateur codé par trellis turbo (TTCQ), reposant sur deux TCQ souples (SOTCQ) est présentée. La séquence de symboles est quantifiée en utilisant un dictionnaire de quantification structuré par le turbo code. La convergence de l'algorithme est étudiée en effectuant un parallèle avec la modulation codée par treillis turbo (TTCM). La méthode de quantification proposée est ensuite étendue au cas du partitionnement de dictionnaires de quantification vectorielle (VQ). Les performances en SNR obtenues en codant des sources i.i.d sans mémoire sont comparées à celles obtenues en utilisant de la quantification scalaire et la TCQ classique.

Introduits dans [114] en 1993, les turbo-codes ont apporté une amélioration considérable de performances des codes correcteurs utilisés en codage de canal, en fournissant une solution pratique approchant de très près la limite de Shannon, tout en conservant une complexité de décodage faible. Ces codes sont construits en utilisant deux codes convolutifs récursifs systématiques mis en parallèle ou en série. Ces deux codeurs convolutifs sont séparés par un entrelaceur de manière à ce que les cycles du graphe de dépendance entre les deux codeurs soient suffisamment longs. Le décodeur procède par inférence probabiliste pour déterminer le message transmis de manière itérative en calculant soit un maximum a posteriori (MAP) [115], soit une approximation telle que l'algorithme de Viterbi souple (SOVA) [116] sur chacun des deux treillis de codes convo-

lutifs constituant le code turbo. L'information *extrinsèque* sur chaque bit utile, calculée à partir de la distribution *a posteriori*, est échangée entre les deux décodeurs convolutifs. La distribution *a posteriori* de ces bits converge après un faible nombre d'itérations (de l'ordre de la dizaine). Le message décodé est finalement obtenu en seuillant les distributions *a posteriori* obtenues en sortie du code convolutif non entrelacé.

Les turbo codes ont également été combinés avec succès avec la modulation codée par trellis (TCM) [117] pour fournir des schémas de modulation très performants. La TCM est une technique de modulation qui fournit un gain de codage tout en conservant le débit et la bande passante utilisée identiques. Pour cela, la constellation d'une modulation sur 2^{n+1} points est partitionnée en deux constellations de 2^n points complémentaires au moyen d'un code convolutif de rendement $n : n+1$. En remplaçant le code convolutif utilisé dans la TCM par un code turbo, des taux d'erreurs bits très faibles sont atteints (10^{-7}) pour un rapport signal à bruit sur le canal seulement supérieur de 1 dB à la limite de Shannon [118]. Dans [119], Robertson et Wörz proposent un schéma similaire à [118] dans lequel les bits de parité sont poinçonnés en alternance à la sortie de chacun des codeurs convolutifs composant le codeur turbo.

La quantification codée par treillis (TCQ) a été introduite dans [120] comme la contrepartie en codage de source de la TCM en codage de canal. Ainsi, de manière similaire, la TCQ repose sur un dictionnaire de source comprenant 2^{n+1} mots de codes, qui est séparé au moyen d'un code convolutif en deux dictionnaires complémentaires comprenant chacun 2^n mots de codes. Le problème du quantificateur est alors de trouver la séquence, correspondant à l'un de ces mots de code, qui minimise la distance euclidienne (donc la MSE) entre la séquence quantifiée et la séquence d'origine. Ainsi la conception d'une TCQ performante consiste à trouver un code convolutif permettant de maximiser la distance entre les mots de codes autorisés.

De la même manière que les turbo codes ont été introduits dans la TCM pour augmenter la distance minimum entre les symboles modulés [119], ces mêmes codes sont considérés ici pour le partitionnement du dictionnaire de source afin d'obtenir une meilleure distribution des valeurs de reconstructions du quantificateur TCQ. Notons que les turbo codes ont également été utilisés pour le codage conjoint source-canal [121] et le codage de sources distribuées avec ou sans pertes [122][123][124]. Cependant, le problème considéré ici est celui de la quantification de sources uniques sans mémoire via l'utilisation de dictionnaires structurés par un turbo code.

Appliquer les turbo codes au problème ci-dessus requiert tout d'abord la conception d'un quantificateur TCQ à sortie souple (SOTCQ). L'algorithme SOTCQ repose sur l'algorithme de Viterbi à sortie souple (SOVA) [116], et des métriques appropriées. La procédure de quantification turbo TCQ peut alors être construite en utilisant deux SOTCQ en alternance. L'information *extrinsèque* sur chaque bit de transition est approximée en estimant l'accroissement ou la diminution de distorsion relative au choix de cette transition (c'est à dire les variations de distorsion induites par le choix de la partition correspondant à la valeur du bit choisie). Le comportement de la TTCQ au cours des itérations est analysé en effectuant un parallèle avec la TTCM. En effet, en supposant un canal à bruit blanc additif gaussien (AWGN), le démodulateur TTCM recherche la meilleure estimation de chaque symbole modulé sachant la séquence bruitée

reçue. Dans le cas du problème de quantification, la séquence originale de symboles peut être vue de manière similaire comme une version bruitée de la séquence des symboles quantifiés. Cependant, une différence importante entre les deux algorithmes réside dans la distribution de ce bruit, qui n'est ni gaussien ni indépendant. L'impact de cette distribution du bruit sur la convergence de l'algorithme est présentée. La procédure de quantification est enfin étendue de manière directe au cas de la quantification TCQ vectorielle. Des résultats de simulations sont présentés dans le contexte de la quantification uniforme de sources uniformes également et sans mémoire ainsi que de sources gaussiennes. Pour des séquences de tailles courtes à moyennes (< 100 symboles), la TCQ turbo offre de meilleures performances que la TCQ classique, avec toutefois une complexité de calcul élevée. Par exemple, un gain de 0.2 dB est obtenu par rapport à la TCQ pour une séquence de 50 échantillons d'une source uniforme quantifiée sur 6 bits/échantillon (le gain maximum théorique étant de 0.4 dB). Pour des séquences plus longues, les performances se dégradent, essentiellement du fait de problèmes de convergence de l'algorithme turbo. La turbo TCQ vectorielle permet toutefois d'obtenir de bonnes performances pour des séquences plus longues.

5.1 TCQ

Soit $\mathsf{X} = (X[1] \dots X[t] \dots X[d])$ une séquence de d symboles à quantifier. Chaque symbole $X[t]$ est supposé indépendant et identiquement distribué selon une loi symétrique notée $\mathbb{P}(X)$. La TCQ [120] est basée sur l'idée de partitionnement d'ensembles proposée par Ungerboeck pour combiner la modulation et le codage de canal [125]. L'approche pour la construction de tels systèmes consiste à partitionner un ensemble initial $\mathcal{C}$ en sous-ensembles complémentaires associés aux transitions entre les états d'un code convolutif (ou, de manière équivalente, aux branches du treillis associé). Pour une longueur de séquence donnée d, l'ensemble des mots de codes valides correspond alors à un sous-ensemble $\mathcal{D}$ de $\mathcal{C}^d$. Ce code est choisi de manière à obtenir des bonnes propriétés de distance (en général euclidienne) sur ce sous-ensemble $\mathcal{D}$. Si $\mathcal{C}$ est une lattice partitionnée en sous-lattices complémentaires, $\mathcal{D}$ forme alors également une lattice, issue du code convolutif. La structure du code permet d'éliminer un grand nombre de points de $\mathcal{C}^d$ (correspondants aux mots de codes invalides) afin d'obtenir le débit désiré. Dans le cas de la quantification, on considère en général pour $\mathcal{C}$ un dictionnaire de quantification de taille modérée (scalaire ou vectoriel). Son partitionnement en sous-dictionnaires est effectué de manière à obtenir des quantificateurs bien adaptés à la source. Nous nous intéresserons ici uniquement au cas des sources sans mémoire (i.e. de loi stationnaire).

Considérons pour l'exemple un dictionnaire de quantification scalaire $\mathcal{C}$ de cardinal 2^{n+1} donné par l'algorithme Lloyd-Max (Fig. 5.1). Ce dictionnaire est partitionné en 4 sous-dictionnaires $\mathcal{C}_0, ..., \mathcal{C}_3$ contenant chacun 2^{n-1} mots de code. Chaque sous-dictionnaire est alors associé à une branche du treillis d'un code convolutif de rendement 1/2 en fonction de la sortie produite par le codeur (Fig. 5.2). Notons que l'association entre la sortie du codeur convolutif et un sous-dictionnaire n'est pas unique. Dans cet exemple, à chaque instant t, seul $\mathcal{C}_0 \bigcup \mathcal{C}_2$ *ou* $\mathcal{C}_1 \bigcup \mathcal{C}_3$ est disponible pour la quantification,

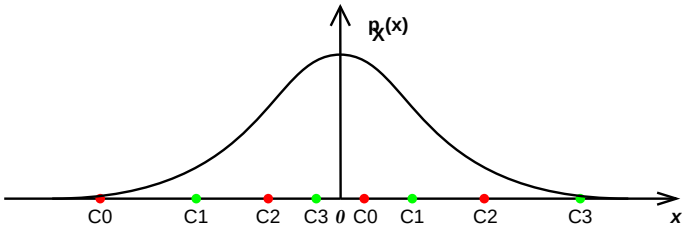

FIG. 5.1: Exemple de partition du dictionnaire initial pour une source gaussienne.

en fonction de l'état du codeur (correspondant aux transitions prises précédemment). Pour obtenir de bonnes performances, il faut donc s'assurer qu'à la fois $\mathcal{C}_0 \bigcup \mathcal{C}_2$ *et* $\mathcal{C}_1 \bigcup \mathcal{C}_3$ forment de bons dictionnaires de quantification pour la source considérée. La distance euclidienne entre les mots de codes autorisés doit donc être maximisée. Notons qu'il est également envisageable d'autoriser certains mots de codes très probables à tout instant, comme proposé dans la TCQ universelle [110] pour les sources gaussiennes et laplaciennes.

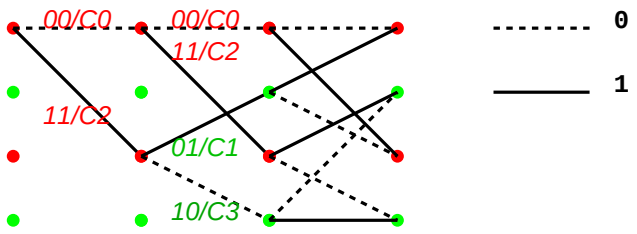

FIG. 5.2: Trellis d'un code convolutif et dictionnaires associés.

La quantification s'effectue en trouvant la séquence minimisant globalement la distance euclidienne entre les séquences permises par le code convolutif et la séquence source à quantifier. L'association entre la sortie du codeur convolutif et les sous-dictionnaires, considérée sur l'ensemble des chemins valides du treillis, définit quels vecteurs d-dimensionnels du dictionnaire produit $\mathcal{C}^d$ sont autorisés.

La séquence de symboles originale peut être également vue comme une version bruitée de la séquence de symboles quantifiée par l'opposé du bruit de quantification. Ainsi, le problème de la TCQ est très similaire à celui de la TCM, où il s'agit de trouver la séquence qui minimise la distance euclidienne entre la séquence reçue et les séquences de symboles de modulation autorisées. Ainsi, un quantificateur TCQ a la même structure qu'un démodulateur TCM, et l'algorithme de Viterbi [126] peut-être appliqué sur le treillis du code convolutif pour trouver la séquence de bits (indexant une séquence permise) qui minimise la distorsion de quantification.

À chaque instant t et pour chaque sous-dictionnaire $\mathcal{C}_i$, le représentant dans $\mathcal{C}_i$ minimisant la distorsion est déterminé par

$$\hat{x}_{\mathcal{C}_i}[t] = \underset{\hat{x}\in\mathcal{C}_i}{\operatorname{argmin}}(\|x[t] - \hat{x}\|^2).$$

Une *métrique de branche* $\rho_i[t] = \|x[t] - \hat{x}_{\mathcal{C}_i}[t]\|^2$ est assignée à chaque branche du treillis labelisée par $\mathcal{C}_i$, comme l'illustre la Fig. 5.1. Cette métrique correspond à la distorsion introduite par la quantification de $x[t]$ dans $\mathcal{C}_i$. L'algorithme de Viterbi est lancé sur le treillis pour sélectionner le chemin minimisant la distorsion totale $\rho = \sum_{t=1}^{d} \|x[t]-\hat{x}[t]\|^2$. Étant donné un état initial arbitraire, l'algorithme de Viterbi produit la séquence de bits B, composée d'un *bit de chemin* par échantillon, et correspondant à la séquence C de sous-dictionnaires utilisée pour la quantification. Ainsi, la TCQ produit un débit fixe total de n bits par échantillon en concaténant le bit de chemin aux bits représentant le mot de code choisi dans le sous-dictionnaire utilisé à cet instant.

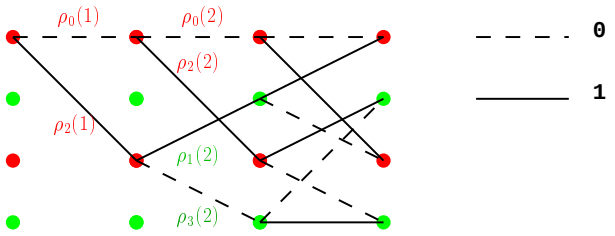

FIG. 5.3: Métriques de branches correspondant à la distorsion.

La structure du déquantificateur est similaire à la structure d'un modulateur TCM. La déquantification est effectuée en codant les bits de chemins B au moyen du code convolutif, permettant ainsi de récupérer la séquence de sous-dictionnaires C. Ensuite, pour chaque instant t, les $n-1$ bits restants sont utilisés pour indexer le mot de code choisi dans $C[t]$.

L'efficacité de la TCQ est liée à la distance euclidienne minimale fournie par le code utilisé, qui peut être améliorée en augmentant la mémoire de ce code. En faisant de la sorte, le nombre d'états du treillis s'accroît également, ce qui augmente la complexité du quantificateur. Quoiqu'il en soit, le processus de déquantification reste extrêmement peu coûteux en temps de calcul.

La TCQ quantifie X avec une complexité algorithmique en $O(d)$ tout en offrant des performances débit-distorsion excellentes. La limite asymptotique du gain dû uniquement à la structure géométrique d'un dictionnaire de quantification vectoriel en dimension d est donnée à haut débit par :

$$D_d^\star(R) = \frac{e(\Gamma(1+\frac{d}{2}))^{\frac{2}{d}} D(R)}{(1+\frac{d}{2})},$$

où le terme $D(R)$ correspond à la fonction débit distortion pour un quantificateur scalaire [120].

Nous ne nous intéressons ici qu'à ce gain structurel dû au choix du code et non au gain dû à l'adéquation des sous-dictionnaires initiaux à la forme de la distribution de la source. Ainsi, nous considérons principalement des sources uniformes. Nous abordons initialement la TCQ scalaire, auquel cas le quantificateur optimal initial, atteignant $D(R)$, est le quantificateur uniforme (dont le dictionnaire forme une lattice). Le cas vectoriel sera traité uniquement en dimension 2.

5.2 TCQ souple

La quantification TCQ extrait la séquence de bits de chemin pour laquelle la distorsion totale est minimale. En d'autres termes, cette technique cherche le vecteur quantifié $\hat{\mathsf{X}}$ le plus proche du vecteur source X au sens de l'erreur quadratique. Cependant, la TCQ ne produit aucune information sur les autres chemins possibles et leur distorsion relative par rapport au chemin optimal. Il pourrait être intéressant de connaître la distorsion relative introduite par le fait de forcer un bit particulier à une valeur spécifique. Ce coût peut être obtenu en remplaçant l'algorithme de Viterbi par un algorithme de Viterbi à sortie souple [116], menant à un algorithme dénommé par la suite TCQ à sortie souple (SOTCQ).

La sortie souple $S[t]$ de l'algorithme de Viterbi à sortie souple est donnée par la différence relative de distorsion entre le chemin de distorsion minimale dont le bit à l'instant t est forcé à 0 et le chemin de distorsion minimale dont le bit à l'instant t est forcé à 1. Ainsi, $S[t]$ peut être écrite comme

$$S[t] = \min_{\substack{\hat{\mathsf{X}}\in\mathsf{C}(\mathsf{B}) \\ \mathsf{B}\in\{0,1\}^d, B[t]=0}} \sum_{s=1}^{d} \|X[s]-\hat{X}[s]\|^2 - \min_{\substack{\hat{\mathsf{X}}\in\mathsf{C}(\mathsf{B}) \\ \mathsf{B}\in\{0,1\}^d, B[t]=1}} \sum_{s=1}^{d} \|X[s]-\hat{X}[s]\|^2,$$

où $\mathsf{C}(\mathsf{B})$ est la séquence de valeurs de reconstructions obtenue en suivant le chemin donné par la séquence de bits B. Le vecteur $\mathsf{B} = (B[1]\dots B[t]\dots B[d])$ de bits de chemin à la sortie de la TCQ pour la séquence X est par conséquent obtenu à partir de la sortie souple de la manière suivante :

$$B[t] = \begin{cases} 1 & \text{si } S[t] > 0, \\ 0 & \text{sinon.} \end{cases} \tag{5.1}$$

Si l'on connaît la distorsion *a priori* introduite par le choix d'un 0 ou d'un 1 à l'instant t, cette information peut être incluse comme information *a priori* $A[t]$ dans l'algorithme de TCQ souple. Si $A[t] > 0$ le choix de $B[t] = 1$ est favorisé par rapport au choix de $B[t] = 0$. De même, forcer la valeur $A[t]$ à $+\infty$ (resp. $-\infty$) correspond à supprimer les branches correspondant au bit 0 (resp. 1) à l'instant t, forçant ainsi $B[t]$ à être égal à 1 (resp. 0). Le signe de $A[t]$ donne le point de quantification respectant la contrainte du treillis tout en étant le plus proche du vecteur source X. L'amplitude de $A[t]$, quant à elle, donne la distance relative entre le vecteur quantifié le plus proche de X, et le vecteur quantifié possédant au moins un bit différent à l'instant t (correspondant en tout à d autres chemins considérés). Cette valeur représente la pénalité

introduite par le choix d'un de ces autres chemins par rapport au choix du chemin optimal. Nous verrons en section 5.3 que ce terme peut être utilisé comme entrée de l'un des quantificateurs composant la turbo TCQ en tant qu'information *extrinsèque*. Dans ce cas, cette information sera formée par la distorsion introduite par l'autre quantificateur composant la turbo TCQ.

L'algorithme SOTCQ fonctionne donc de la façon suivante. L'état initial est mis arbitrairement à zéro. Ainsi, la métrique de distorsion avant $\alpha_0[1]$ est mise à 0 pour l'état zéro, tandis qu'elle est mise à $\alpha_j[1] = +\infty$ pour tout autre état j tel que $j \neq 0$. L'algorithme effectue ensuite une récursion avant en calculant les métriques d'états :

$$\alpha_j(t+1) = \min(\alpha_{j_0}[t] + \rho_{j_0,j}[t], \alpha_{j_1}[t] + \rho_{j_1,j}[t]), \tag{5.2}$$

où j_b est le noeud connecté à j par la branche indexée par le bit b, et $\rho_{j_b,j}[t]$ est la distorsion assignée à cette branche. La récursion arrière s'effectue de manière similaire en initialisant tout d'abord les métriques d'état $\beta_j[d+1]$ à 0 pour tout j. Elle sont ensuite calculées récursivement de la façon suivante :

$$\beta_j[t-1] = \min(\beta_{j_0}[t] + \rho_{j,j_0}(t-1), \beta_{j_1}[t] + \rho_{j,j_1}[t-1]). \tag{5.3}$$

La sortie souple se dérive alors de l'expression suivante :

$$S[t] = \min_j(\alpha_j[t] + \rho_{j,j_1}[t] + \beta_{j_1}[t+1]) - \min_j(\alpha_j[t] + \rho_{j,j_0}[t] + \beta_{j_0}[t+1]).$$

Notons que dans le cas où un code systématique est utilisé, les bits systématiques peuvent être poinçonnés afin d'ajuster le débit. Les métriques de distorsion $\rho_i[t]$ de toutes les branches sont alors mises à zéro aux instants t correspondant aux instants où les bits systématiques ont été poinçonnés.

5.3 Turbo TCQ

Le problème traité dans cette section est celui de l'utilisation de turbo codes pour le partitionnement du dictionnaire de quantification, afin d'augmenter la distance euclidienne entre les séquences quantifiées, permettant d'obtenir une meilleure couverture de l'espace et minimisant ainsi la distorsion. Une TCQ basée sur des turbo codes peut être construite en plaçant deux TCQ souples en parallèle, comme illustré figure 5.4. La structure de la turbo TCQ est alors très similaire à la structure d'un décodeur turbo TCM [119]. Pour les expériences conduites dans ce chapitre, un code systématique récursif de rendement 1/2 a été utilisé pour les deux quantificateurs TCQ souples. L'une des TCQ souple ($\mathcal{A}$) est appliquée directement sur la séquence X tandis que l'autre ($\mathcal{B}$) est appliquée sur la séquence entrelacée $\tilde{\mathsf{X}}$. Ici nous considérons également un quantificateur scalaire sur 2^{n+1} niveaux qui sera partitionné en 4 dictionnaires (du fait du rendement du code convolutif). Pour obtenir un rendement total de 1/2, les symboles source sont

poinçonnés aux instants impairs dans le treillis de la TCQ souple $\mathcal{A}$ et aux instants pairs dans le treillis de la TCQ souple $\mathcal{B}$. Ainsi, les métriques de branches des treillis $\mathcal{A}$ et $\mathcal{B}$ sont données par :

$$\begin{cases} \rho_k^{\mathcal{B}}[t] = 0, \rho_k^{\mathcal{A}}[t] = \|X[t] - \hat{X}_{C_k}[t]\|^2 \text{ si } t \text{ est pair} \\ \rho_k^{\mathcal{A}}[t] = 0, \rho_k^{\mathcal{B}}[t] = \|X[I[t]] - \hat{X}_{C_k}[I[t]]\|^2 \text{ si } t \text{ est impair.} \end{cases}$$

Les métriques aux instants correspondant à des symboles sources poinçonnés sont mises à zéro. Afin de garder les symboles des instants pairs à des positions paires et les symboles des instants impairs à des positions impaires, un entrelaceur pair-impair est utilisé. Nous avons de plus choisi un entrelaceur de type S-aléatoire (respectant un espace minimum de S instants entre les positions entrelacées) pour éviter les cycles courts dans le graphe de dépendance du turbo code. L'entrelacement est ainsi défini comme une fonction bijective :

$$\begin{aligned} I : [\![1, d]\!] &\to [\![1, d]\!] \\ t &\mapsto \tilde{t}, |t - \tilde{t}| > S, t + \tilde{t} \in 2\mathbb{N}. \end{aligned}$$

5.3.1 Quantification

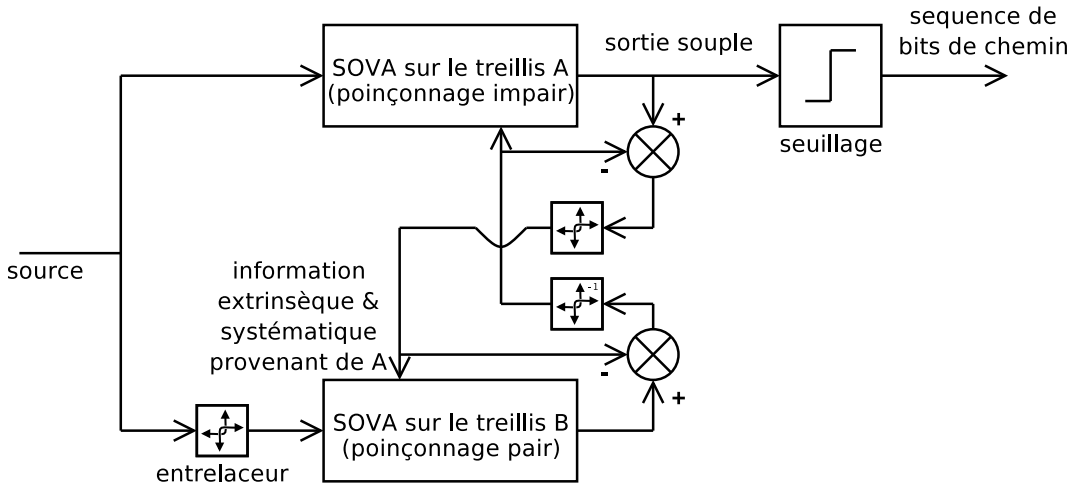

FIG. 5.4: Structure du quantificateur.

En utilisant l'entrelacement et le poinçonnement décrits ci-dessus, les observations à l'entrée de la TTCQ sont reliées aux observations à l'entrée d'un turbo code binaire obtenu à partir de deux codeurs convolutifs récursifs systématiques de rendement 1/2 en poinçonnant la sortie de l'un ou l'autre codeur alternativement. En revanche, contrairement au cas des turbo codes binaires, l'observation correspondant au bit de parité conservé et celle correspondant aux bits systématiques est couplée car ces bits sont utilisés ensemble pour désigner un symbole quantifié. C'est la raison pour laquelle, de même

que pour la TTCM, l'information qui est échangée entre les quantificateurs composant la TTCQ est définie de manière légèrement différente que dans le cas du décodage itératif des turbo codes binaires. Plus précisément, la sortie souple de chaque TCQ souple est représentée sous la forme d'une somme d'information *systématique et extrinsèque* et d'une information *a priori* $S[t] = E[t] + A[t]$. Dans le cas des turbo codes binaires, il est plus usuel de séparer cette même information en trois parties : extrinsèque, a priori et systématique. Ici, il est impossible de séparer l'information systématique de l'information extrinsèque [119]. Ceci n'est toutefois pas gênant car du fait du poinçonnement, l'information systématique n'est utilisée qu'une seule fois dans chaque TCQ souple, tandis que l'information extrinsèque est échangée entre ces deux quantificateurs.

La quantification se déroule alors de manière itérative en effectuant les quantifications TCQ souples alternativement, en commençant par la TCQ souple $\mathcal{A}$. Avant la première itération, l'information *a priori* de la TCQ souple $\mathcal{A}$ peut être initialisée à 0 pour tout instant de sorte que tout chemin soit équiprobable. Alternativement, il est possible d'extraire une information *a priori* sur les bits de chemins aux instants $t^\star$ correspondants aux symboles poinçonnés dans la TCQ souple $\mathcal{A}$, à partir des observations à l'entrée de la TCQ souple $\mathcal{B}$. Cette information *a priori* prend en compte la pénalité initiale sur la distorsion induite par le choix d'une valeur particulière pour chaque bit $B[t^\star]$. En effet, à cause du poinçonnement, l'impact de ces bits sur la distorsion totale ne serait pas considéré pendant la première minimisation dans le treillis $\mathcal{A}$. Ainsi, comme dans [119], les métriques *a priori* pour les instants $t^\star$ de la TCQ souple $\mathcal{A}$ peuvent être initialisées à la différence locale en distorsion entre la transition de distorsion minimale correspondant à $B[t^\star] = 0$ et celle correspondant à $B[t^\star] = 1$. Spécifiquement, en utilisant l'association entre les bits de transition et les sous-dictionnaires proposée et illustrée figure 5.2, où $B[t^\star] = 0$ correspond à $\mathcal{C}_0$ ou $\mathcal{C}_3$ et $B[t^\star] = 1$ correspond à $\mathcal{C}_1$ ou $\mathcal{C}_2$, l'information *a priori* est initialisée comme suit :

$$\begin{cases} A_0[t] = 0 \text{ si } t \text{ est pair,} \\ A_0[t] = \min\limits_{k=0,3} \rho_k^B[I^{-1}[t]] - \min\limits_{k=1,2} \rho_k^B[I^{-1}[t]] \text{ si } t \text{ est impair.} \end{cases}$$

A l'itération j, chaque TCQ souple produit la séquence souple S_j. La distorsion *a priori* A_j est ensuite soustraite de la sortie souple S_j pour obtenir l'information extrinsèque E_j. Le terme $E_j[t]$ correspond à la distorsion additionnelle introduite en forçant le bit $B[t]$ à 0 au lieu de 1 (ou 1 au lieu de 0 si $E_j[t]$ est négative) dans le quantificateur souple considéré. L'information extrinsèque est alors entrelacée (resp. désentrelacée) par I (resp. I^{-1}) et fournie comme information *a priori* en entrée de l'autre TCQ souple $\mathcal{B}$ (resp. $\mathcal{A}$). Ce processus est itéré jusqu'à ce que la sortie souple des deux quantificateurs converge, ou jusqu'à qu'un nombre maximum d'itérations fixé soit atteint. La séquence finale de bits de chemin est obtenue en seuillant la sortie souple du quantificateur $\mathcal{A}$.

De plus, $n-1$ bits par échantillons sont utilisés, soit pour indexer le mot de code $\hat{X}_{C_k[t]}$ dans le quantificateur $\mathcal{A}$ aux instants t pairs, soit pour indexer le mot de code $\hat{X}_{C_k[I[t]]}$ dans le treillis $\mathcal{B}$ aux instants t impairs. Le mot de code de turbo TCQ final est obtenu à chaque instant en concatenant ces bits au bit de chemin, résultant en un débit total fixe de n bits par échantillon.

5.3.2 Déquantification

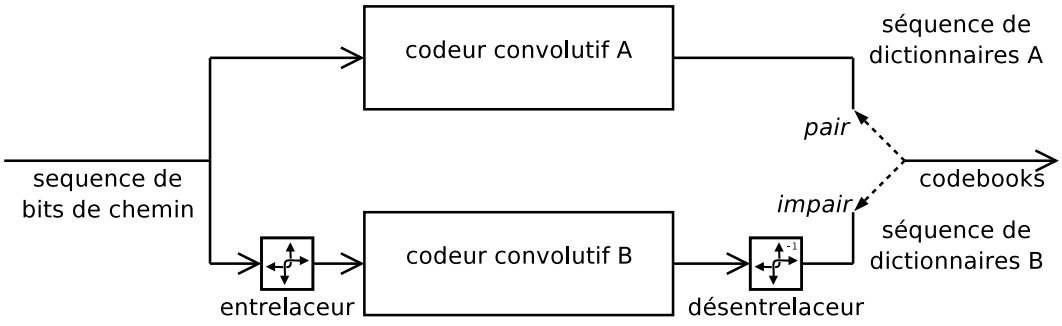

FIG. 5.5: Structure du déquantificateur.

Le déquantificateur est constitué de deux codeurs convolutifs récursifs systématiques en parallèle. Le codeur $\mathcal{A}$ lit le flux de bits de chemin $\tilde{\mathsf{B}}$ directement tandis que le codeur $\mathcal{A}$ en lit une version $\tilde{\mathsf{B}}$ préalablement entrelacée par I. À cause du poinçonnement, seule l'une des sorties des deux codeurs est valide à chaque instant t. Pour les instants pairs, la sortie du codeur $\mathcal{A}$ est utilisée pour retrouver le sous-dictionnaire $C_k[t]$ utilisé lors de la quantification. Pour les instants impairs, la sortie du codeur $\mathcal{B}$ est utilisée pour retrouver la séquence de sous-dictionnaires qui est ensuite désentrelacée par I^{-1}. Ce processus est illustré figure 5.5. À chaque instant, les $n-1$ bits restants sont utilisés pour indexer le sous-dictionnaire $C_k[t]$ et retrouver la valeur de reconstruction appropriée.

5.4 Analyse de la convergence

Contrairement à l'algorithme de Viterbi utilisé dans la TCQ, qui extrait le centroïde le plus proche de l'observation indépendamment de sa distribution, la turbo TCQ repose sur l'algorithme turbo itératif dont la convergence peut échouer dans certains cas. Nous étudions ici le comportement de cet algorithme dans le cadre de la turbo TCQ en s'appuyant sur le parallèle existant entre la turbo TCQ et la turbo TCM. Les résultats expérimentaux indiquent qu'il existe d'importantes différences dans le comportement de l'algorithme itératif dans le cas du codage de source par rapport au cas du codage de canal.

En turbo TCM, pour des rapports signal sur bruit supérieurs à un certain seuil, l'algorithme converge en un petit nombre d'itérations (une dizaine) vers un point fixe qui coïncide quasiment systématiquement avec le vecteur transmis. En turbo TCQ, la probabilité que l'algorithme converge est significativement plus faible que dans le cas de la TTCM et la vitesse de convergence est très lente. De plus, à mesure que la longueur de la séquence augmente, la proportion de séquences pour lesquelles l'algorithme converge décroît. Il est cependant observé que lorsque cette proportion est raisonnable, la turbo TCQ offre de meilleures performances que la TCQ. Ceci apparaît pour des séquences de taille relativement courte, de moins d'une centaine de symboles.

Une différence majeure entre la turbo TCM à modulation d'amplitude et la turbo TCQ réside dans la distribution du vecteur aléatoire X sur lequel l'algorithme itératif de décodage (resp. de quantification) est appliqué. Considérons la TTCQ d'une source uniforme sans mémoire pour des raisons de simplicité. Des vecteurs d symboles source $X[t]$ indépendants et uniformément distribués dans l'intervalle $[-\frac{a}{2}, \frac{a}{2}]$ sont quantifiés sur n bits par symbole. Il en résulte un total de 2^{nd} mots de code de TTCQ résidant à l'intérieur d'un hypercube de coté a. Ce quantificateur est comparé à la TTCM basée sur une modulation d'amplitude à 2^n niveaux, et où les séquences émises sont équiprobables et de longueur d. Ces deux systèmes partagent le même dictionnaire $\mathcal{C} = \{\hat{\mathsf{X}} \in \mathbb{R}^d\}$ représentant soit les séquences modulées, soit les centroïdes de quantification.

Dans la turbo TCM, un mot de code $\hat{\mathsf{X}}$ est émis par le modulateur dans l'espace vectoriel $\mathbb{R}^d$. Le turbo code permet alors d'obtenir de bonnes propriétés de distance minimale entre les mots de codes tout en étant décodable de manière itérative à faible complexité. Dans un canal à bruit blanc gaussien additif (AWGN), chaque composante $\hat{X}[t]$ du symbole modulé est perturbée indépendamment par un bruit additif gaussien de variance σ^2. L'utilisation d'un algorithme SOVA dans le décodeur turbo revient alors à trouver la séquence $\hat{\mathsf{x}}$ la plus probable *a posteriori* par rapport à l'observation de la séquence bruitée x_c (MAP séquence).

Dans la turbo TCQ, un vecteur $\mathsf{X_s}$ est initialement distribué uniformément dans un hypercube de l'espace vectoriel $\mathbb{R}^d$. Le but est alors de trouver le vecteur $\hat{\mathsf{x}}$ le plus proche de x_s. Puisque chaque symbole $X_s[t]$ composant X_s est tiré uniformément dans l'intervalle $[0, a]$, la densité de probabilité du vecteur source X_s est constante et égale à $\frac{1}{a^d}$ sur un hypercube de coté a, et s'annule en dehors. Le dictionnaire $\mathcal{C}$ formant une lattice du fait du code linéaire utilisé, toutes les cellules de Voronoï sont identiques et de volume $\frac{a^d}{2^{nd}}$ hormis un nombre très faible de cellules situées aux bords de l'hypercube.

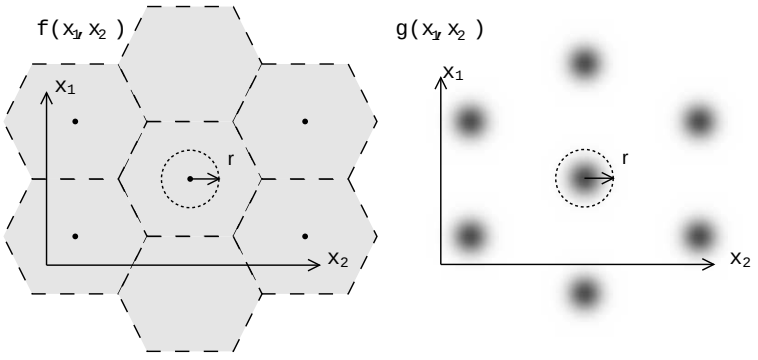

FIG. 5.6: Comparaison entre la distribution de X_s et X_c pour la TCQ uniforme et pour la TCM par amplitude sur un canal AWGN respectivement, en dimension 2.

Dans les deux cas, le vecteur aléatoire sur lequel l'algorithme itératif est appliqué peut être modélisé de la même façon, en posant $\mathsf{X_c} = \hat{\mathsf{X}} + \mathsf{N_c}$ et $\mathsf{X_s} = \hat{\mathsf{X}} + \mathsf{N_s}$. Cependant la différence entre les deux systèmes réside dans la distribution de N_c et N_s. La figure 5.6 montre dans quelle mesure les distributions de X_s et X_c diffèrent dans le cas de la TCQ

d'une source uniforme et de la TCM sur un canal AWGN (pour $d = 2$). La distribution du bruit dans chaque cas peut s'écrire de la manière suivante :

$$\mathbb{P}(\mathsf{N}_s) = \begin{cases} \frac{2^{nd}}{a^d} \text{ si } \mathsf{n}_s \text{ est à l'intérieur d'une cellule de Voronoï centrée en l'origine} \\ 0 \text{ sinon,} \end{cases}$$

$$\mathbb{P}(\mathsf{N}_c) = \frac{1}{(2\pi\sigma^2)^{\frac{d}{2}}} e^{-\frac{\|\mathsf{n}_c\|^2}{2\sigma^2}}.$$

Examinons à présent comment cette différence dans la distribution du bruit peut influencer le comportement de l'algorithme turbo. Comme modèle simple du comportement en convergence de l'algorithme nous supposons qu'un point fixe est atteint et correspond à $\hat{\mathsf{X}}$ si et seulement si l'observation X_c ou X_s se trouve à une distance r_C du vecteur $\hat{\mathsf{X}}$. Dans ce modèle simplifié, nous ne considérons pas la possibilité d'avoir un point fixe différent de $\hat{\mathsf{X}}$, ainsi si l'observation se trouve en dehors de l'hypersphère de rayon r_C centrée sur $\hat{\mathsf{X}}$, nous supposons que l'algorithme ne converge pas. On peut alors calculer les probabilités $\mathbb{P}(\|\mathsf{X}_c - \hat{\mathsf{X}}\| < r)$ et $\mathbb{P}(\|\mathsf{X}_s - \hat{\mathsf{X}}\| < r)$ d'être proche de la solution pour la TTCQ et la TTCM, au même rapport signal à bruit. Dans l'annexe 5.10 nous donnons une preuve que ces probabilités sont telles que :

$$\mathbb{P}(\|\mathsf{X}_s - \hat{\mathsf{X}}\| < r) < (\frac{\pi}{6})^{\frac{d}{2}} \mathbb{P}(\|\mathsf{X}_c - \hat{\mathsf{X}}\| < r),$$

pour un rayon r petit et à haut débit. En d'autres termes, la probabilité que le vecteur source soit proche d'un centroïde de la TTCQ pour la quantification d'une source uniforme sans mémoire décroit exponentiellement au fur et à mesure que la longueur de la séquence augmente par rapport à la probabilité que la séquence bruitée reçue soit proche de la séquence émise dans le cas la TTCM par modulation d'amplitude. Ceci indique qu'on peut s'attendre à des performances bien plus médiocres de l'algorithme TTCQ par rapport à l'algorithme TTCM en termes de taux de convergence, en particulier en grande dimension d. Notons que pour des sources gaussiennes ou laplaciennes (et de manière générale n'importe quelle distribution stationnaire non singulière) le comportement de l'algorithme est similaire. En effet, pour un débit suffisamment élevé (correspondant à une distorsion raisonnable sur la source), la densité de probabilité à l'intérieur d'une cellule de Voronoï est approximativement constante.

5.5 Interprétation géométrique de la convergence

Nous proposons ici une interprétation complémentaire du problème de convergence de l'algorithme turbo reposant sur des arguments de théorie de l'information et de géométrie.

Examinons tout d'abord le cas de la TTCM pour une séquence émise à une puissance σ_S^2 sur un canal à bruit blanc additif gaussien (AWGN) de variance σ_N^2. Pour s'assurer

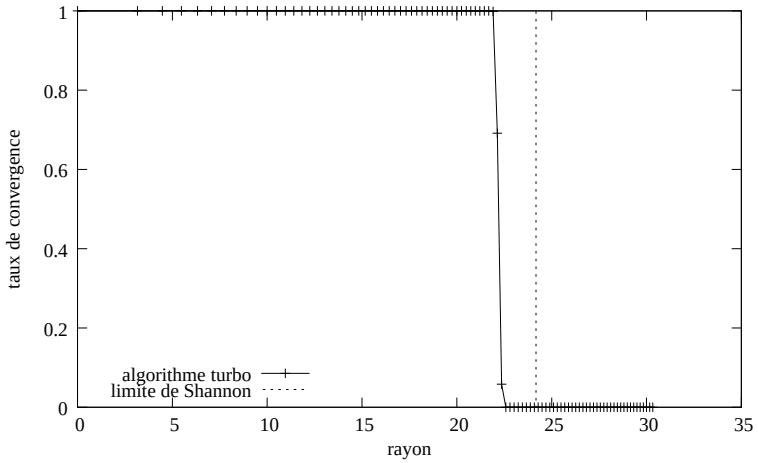

FIG. 5.7: Taux de convergence de l'algorithme turbo en fonction de la distance au centroïde pour une séquence de 10000 échantillons et des codes $(5, 7)$. Le rayon du pavage de sphères le plus dense associé en dimension 10000, correspondant à la limite de Shannon, est donné par $r_L = \frac{1}{\sqrt{\pi}}\Gamma(1+\frac{d}{2})^{\frac{1}{d}} = 24.2096$.

d'une transmission correcte de la séquence, le théorème de Shannon dit que le débit ne doit pas excéder la capacité du canal, ce qui donne une borne supérieure du nombre de mots de codes pouvant être utilisés [1] :

$$2^{nd} < (1 + \frac{\sigma_S^2}{\sigma_N^2})^{\frac{d}{2}}. \tag{5.4}$$

A mesure que d tend vers l'infini, la meilleure manière de placer les mots de codes dans l'espace correspond à la solution du problème d'empilement de 2^{nd} hypersphères de rayon $r = \sqrt{d\sigma_N^2}$ dans l'hypersphère de rayon $R = \sqrt{d(\sigma_S^2 + \sigma_N^2)}$. L'équation 5.4 est réécrite pour faire apparaître le rapport entre les volumes de ces hypersphères :

$$2^{nd} < \frac{V_d R^d}{V_d r^d}, \tag{5.5}$$

où $V_d = \pi^{\frac{d}{2}}/\Gamma(1+\frac{d}{2})$ est le volume d'une hypersphère unitaire en dimension d. À haut débit, si le signal est restreint à l'hypercube de coté a plutôt que l'hypersphère de rayon R, le nombre de mots de code contenus dans cet hypercube est donné par :

$$2^{nd} < \frac{a^d}{V_d r^d}. \tag{5.6}$$

Ainsi, un signal émis avec la TTCM considérée sur un canal AWGN peut être décodé correctement uniquement s'il réside à l'intérieur d'une hypersphère de rayon r_L autour de la séquence modulée, où r_L est défini par :

$$r_L = \frac{a}{2^n}\frac{1}{\sqrt{\pi}}\Gamma(1+\frac{d}{2})^{\frac{1}{d}}. \tag{5.7}$$

Pour la TTCQ, le but est de trouver le vecteur $\hat{\mathsf{X}}$ le plus proche du vecteur source X_s distribué aléatoirement dans l'espace vectoriel. La source étant sans mémoire et uniforme, la distribution du vecteur X est constante sur l'hypercube de coté a et égale à $1/a^d$. En négligeant à nouveau les effets aux bords de l'hypercube, toutes les cellules de Voronoï de la TTCQ sont congrues à un polyèdre de volume $a^d/2^{nd}$. Ce polyèdre est approximativement sphérique en dimension élevée, de rayon limite r_L donné par l'égalité des volumes $V_d r_L^d = a^d/2^{nd}$.

La TTCQ et la TTCM diffèrent uniquement au niveau de la distribution du vecteur source X_s et du vecteur reçu X_c sur lesquels l'algorithme turbo est appliqué. Nous supposons ici que l'algorithme converge vers un point fixe si sa sortie reste inchangée pendant plus de 5 itérations. Considérons des séquences de $d = 10000$ échantillons codées à $n = 8$ bits/échantillon dans un hypercube de coté $a = 256$. Pour 20 itérations et un turbo code basé sur un code convolutif systématique récursif de polynômes générateurs $(5, 7)$, les résultats expérimentaux montrent que l'algorithme n'arrive pas à converger vers le vecteur le plus proche $\hat{\mathsf{X}}$ avec une probabilité inférieure à 10^{-3} lorsque la distance à l'observation X est inférieure à $r_C = 22.1$ (Fig. 5.7). Pour ces paramètres, le rayon limite est donné par $r_L = 24.21$. Dans le cas de la TTCM, en appliquant le théorème central limite au vecteur X bruité par un bruit gaussien indépendant de variance σ_N^2, il apparaît que la distribution des distances de $\hat{\mathsf{X}}$ à X est fortement centrée sur $r = \sqrt{d\sigma_N^2}$. Ainsi, $\hat{\mathsf{X}}$ se trouve dans le rayon de convergence de l'algorithme turbo dès que $r < r_C$. Cela signifie que pour un rapport signal sur bruit supérieur de seulement $10\log_{10}(\frac{r_L^2}{r_C^2}) = 0.7dB$ à la limite de Shannon, la probabilité que le message soit décodé correctement est très forte. Dans le cas de la TTCQ par contre, le vecteur source X a une probabilité proche de 1 d'être en dehors de l'hypersphère de rayon r_C du fait de sa distribution uniforme. En effet, le rapport entre le volume de l'hypersphère de rayon $r_C < r_L$ et la cellule de Voronoï décroit exponentiellement lorsque la dimension d augmente. Même avec un grand nombre d'itérations, l'algorithme turbo échoue en dimension élevée, ce qui mène à de mauvaises performances du quantificateur pour des séquences longues.

5.6 Adaptation de l'algorithme TTCQ aux cas d'échecs

Il n'est pas surprenant de constater que le gain apporté par la TTCQ sur la TCQ en termes d'erreur de reconstruction s'accroît avec le taux de convergence de l'algorithme turbo sous-jacent. Nous supposons dans nos expériences que l'algorithme a convergé vers un point fixe si la séquence binaire représentant le mot de code en sortie de l'algorithme TTCQ est inchangée pendant plus de 50 itérations. Si l'algorithme ne converge pas après un grand nombre d'itérations, la séquence est dite 'non-convergente'. Dans cette section nous présentons plusieurs modifications de l'algorithme afin d'améliorer la probabilité de convergence ou, le cas échéant, d'obtenir une séquence quantifiée de distorsion minimale.

Il a été observé que pour les séquences non-convergentes, la distribution des métriques *a posteriori* reste centrée en zéro (Fig. 5.8), ce qui provoque des changements significatifs de la sortie à chaque itération. Dans ce cas, la sortie finale B correspond à la séquence de distorsion minimale parmi les séquences testées à chaque itération :

$$\mathsf{B} = \underset{\mathsf{H}_j, j \in [\![1,N]\!]}{\operatorname{argmin}} \sum_{t=1}^{d} \|X(t) - \hat{X}_{\mathsf{C}(\mathsf{H}_j)}(t)\|^2$$

Il s'agit là d'une autre différence fondamentale avec la turbo TCM pour laquelle la séquence émise $\hat{\mathsf{X}}$ n'est bien entendu pas accessible. On peut alors voir l'algorithme TTCQ comme une méthode de recherche en temps linéaire ($O(d)$) parmi un dictionnaire structuré en dimension finie, y compris dans les cas ou l'algorithme ne converge pas.

Nous avons évoqué dans la section 5.1 que l'association entre les sous-dictionnaires et la sortie du code turbo binaire est arbitraire tant que les centroïdes de quantification autorisée à chaque instant (i.e. $\mathcal{C}_0 \bigcup \mathcal{C}_2$ ou $\mathcal{C}_1 \bigcup \mathcal{C}_3$ pour un code de rendement 1/2) forment de bons quantificateurs de la source. La distance entre chaque centroïde doit donc être maximisée. Le tableau 5.1 montre quatre possibilités d'association de la sortie du turbo code aux sous-dictionnaires satisfaisant toutes cette condition de distance maximale. Par exemple, pour l'association A, les sorties 00, 01, 11 et 10 du code turbo sont associées aux sous-dictionnaires $\mathcal{C}_0$, $\mathcal{C}_1$, $\mathcal{C}_2$ et $\mathcal{C}_3$ respectivement. Ces quatres associations différentes mènent à quatre dictionnaires vectoriels différents correspondant aux chemins permis par la structure du treillis. Ainsi, bien que chaque association mène aux mêmes performances moyennes, une association pourrait être plus appropriée à une réalisation particulière de la séquence de symboles à quantifier. En effet, une réalisation x est plus proche de l'un des dictionnaires vectoriels, résultant en une probabilité de convergence vers un point fixe accrue. L'association correspondant peut ainsi être retenue et transmise au décodeur. Cependant ceci provoque un surcoût en temps de calcul au niveau du quantificateur.

L'envoi de l'association séléctionnée au décodeur entraîne un surcoût en débit de 1 ou 2 bits par séquence transmise, en fonction du nombre d'associations autorisée choisi (soit A et B, soit A, B, C et D). Par la suite, nous nous référons à la turbo TCQ utilisant uniquement les associations A et B sous le nom de DTTCQ, tandis que la QTTCQ correspondra à la turbo TCQ s'autorisant les quatres associations. Les résultats expérimentaux sur le taux de convergence et la performance en distorsion pour ces différents paramètrages de la turbo TCQ sont présentés dans la section 5.8. Nous utilisons également la même association dans les deux treillis des codes convolutifs sous-jacents.

Une autre manière de voir le problème de convergence est de considérer l'algorithme turbo comme une instance particulière de l'algorithme de passage de messages de Pearl [127]. Cet algorithme effectue l'inférence probabiliste sur un graphe où chaque noeud représente une variable aléatoire et les branches (non orientées) représentent les dépendances statistiques entre ces variables. La convergence de cet algorithme est garantie sur les arbres. Cependant, il a été appliqué avec succès dans des graphes cycliques tels que les graphes de turbo codes et plus récemment de codes LDPC (low-density parity check

sortie du code	association A	B	C	D
00	$\mathcal{C}_0$	$\mathcal{C}_1$	$\mathcal{C}_2$	$\mathcal{C}_3$
01	$\mathcal{C}_1$	$\mathcal{C}_0$	$\mathcal{C}_3$	$\mathcal{C}_2$
11	$\mathcal{C}_2$	$\mathcal{C}_3$	$\mathcal{C}_0$	$\mathcal{C}_1$
10	$\mathcal{C}_3$	$\mathcal{C}_2$	$\mathcal{C}_1$	$\mathcal{C}_0$

TAB. 5.1: Association entre le code binaire de rendement 1/2 et les sous-dictionnaires

codes) [128] [129]. Bien que les hypothèses de convergence ne soient pas respectées, les messages (correspondant aux rapports de vraisemblances dans le cas des codes turbo ou LDPC) sont suffisamment différents pour que les lois sur chaque noeud convergent avant que ces messages n'effectuent trop de cycles. Dans le cas de la turbo TCQ par contre, les rapports de vraisemblances sont très proches (un grand nombre de centroïdes correspondent à des distorsions similaires). Les approches consistant à modifier l'algorithme, comme suggéré dans [130], ou à utiliser d'autres méthodes de propagation de croyance [131] n'ont pas donné de résultats satisfaisants lorsque nous les avons appliquées au cas de la turbo TCQ.

5.7 Turbo TCQ vectorielle

La TCQ a été étendue à la quantification vectorielle dans [132]. De manière similaire, il est possible d'utiliser des sous-dictionnaires de quantification vectorielle pour la turbo TCQ. Le problème est alors de construire un partitionnement approprié du quantificateur vectoriel initial en lattices (dans le cas de la quantification de sources uniformes) dans un espace de dimension q. Dans ce cas, le dictionnaire $\mathcal{C}^p$ est partitionné par le turbo code, où $\mathcal{C}$ est un dictionnaire vectoriel de dimension q. Pour une séquence de longeur $d = pq$, les échantillons d'entrée $X[t]$ sont groupés en vecteurs de dimension q, définis par :

$$\mathsf{X}_q[t] = (\mathsf{X}[qt + k - q])_{k \in [\![1,q]\!]}.$$

La structure du quantificateur reste la même, excepté pour la longueur du treillis qui est égale à p. Le dictionnaire de quantification vectorielle $\mathcal{C}$ est séparé en quatre ensembles $\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3$ de sorte que $\mathcal{C}_0 \bigcup \mathcal{C}_2$ et $\mathcal{C}_1 \bigcup \mathcal{C}_3$ forment de bon quantificateurs vectoriels pour la source considérée. Notons que $\mathcal{C}$ lui-même n'a pas besoin d'être le meilleur quantificateur vectoriel de cette source. Dans le cas de la quantification uniforme, $\mathcal{C}_0 \bigcup \mathcal{C}_2$ et $\mathcal{C}_1 \bigcup \mathcal{C}_3$ sont des classes d'équivalences de $\mathcal{C}$. Les métriques de distorsion sont calculées dans l'espace de dimension q par $\rho_i[t] = \|\mathsf{X}_q[t] - \hat{\mathsf{X}}_{q,\mathcal{C}_i}[t]\|^2$. Afin de comparer les performances de la quantification TTCQ vectorielle et scalaire au même débit, le nombre de centroïdes dans $\mathcal{C}$ est fixé à la valeur 2^{qn+1}. L'algorithme itératif de quantification est appliqué de manière identique au cas scalaire. Comme cet algorithme reste identique, avec une longueur de treillis réduite d'un facteur q, et un bruit de quantifi-

cation plus faible pour chaque vecteur, on peut s'attendre à de meilleures performances de la TTCVQ par rapport à la TTCQ pour des séquences longues. Comme pour le cas scalaire, un changement d'associations des sorties du code convolutif aux dictionnaires, menant aux algorithmes dénommés DTTCVQ (pour 2 associations) et QTTCVQ (pour 4 associations), permet d'accroître les performances.

La figure 5.9 illustre la partition de l'espace 2D pour une quantification vectorielle uniforme bidimensionnelle, où le dictionnaire $\mathcal{C}$ est le produit de deux dictionnaires de quantification scalaire uniforme. Bien que $\mathcal{C}$ ne soit pas le dictionnaire de quantification optimal pour cette source, il est partitionné en deux lattices hexagonales $\mathcal{C}_0 \bigcup \mathcal{C}_2$ et $\mathcal{C}_1 \bigcup \mathcal{C}_3$, qui sont connues pour être la structure de quantification optimale de cette source en 2D. Comme pour la TCQ scalaire, la grille disponible à un instant donné pour la quantification dépend des choix effectués pour la quantification des vecteurs précédents.

Il existe de nombreuses autres extensions de la quantification TCQ qui pourraient être appliquées à la quantification TTCQ. En particulier, la quantification TCQ universelle [110] est bien adaptée à la quantification de sources symétriques centrées de densité de probabilité décroissante sur les abcisses positives. Elle consiste à considérer un dictionnaire de quantification scalaire dont seule la position des premiers centroïdes est transmise au décodeur, les autres étant répartis uniformément dans l'intervalle. Le centroïde nul est également conservé dans les deux partitions du dictionnaire afin de quantifier efficacement cette valeur plus probable. Bien que nous ayons considéré uniquement des sources uniformes pour lesquelles cette technique n'a pas d'intérêt, elle pourrait s'adapter sans mal à la turbo TCQ.

Il est également possible d'étendre la turbo TCQ en vue d'un couplage avec un codeur entropique. La turbo TCQ contrainte en entropie consiste alors à effectuer les mêmes traitements que pour la turbo TCQ classique pour une métrique différente prenant en compte le débit obtenu après codage entropique. En effet, on cherche alors le compromis débit-distorsion optimal en remplaçant la métrique de distortion ρ par une métrique $\rho + \lambda R$ où R est le coût de codage du mot de code correspondant à la branche considérée et λ représente l'opposé de la pente sur la courbe débit-distorsion. Une recherche itérative de la valeur de λ correspondant au débit cible désiré fournit alors le chemin offrant la distorsion minimale à ce débit. À nouveau, dans le cas d'une source uniforme, le débit obtenu est identique quelque soit le chemin choisi dans le treillis, et nous n'avons donc pas appliqué cette technique ici.

5.8 Résultats de simulation

Les performances débit-distorsion de la TTCQ sont comparées à celles de la TCQ et de la quantification scalaire pour des sources uniformes et gaussiennes. Les codes convolutifs composant le turbo code sont des codes récursifs systématiques de polynômes générateurs $(5, 7)$. Le rendement de ce code est de $1/2$, et il forme un treillis à 4 états. L'entrelaceur utilisé dans la turbo TCQ est un entrelaceur S-aléatoire fixé pour toutes les expériences. Les résultats présentés sont moyennés sur 10^5 expériences. À moins que

cela ne soit précisé explicitement, des séquences de 50 échantillons sont quantifiées sur 8 bits par échantillons, avec au plus 10^4 itérations de l'algorithme turbo par séquence. Dans la première expérience, les échantillons $X[t]$ de la séquence source sont distribués uniformément, et les sous-dictionnaires $\mathcal{C}_i$ de la turbo TCQ sont uniformes également, de même pas de quantification. Le tableau 5.2 montre la performance débit-distorsion de la TTCQ comparée à la TCQ à différents débits pour des séquences courtes de 50 échantillons. La borne inférieure (pour que ces quantifications aient un itérêt) est donnée par la fonction débit-distorsion du quantificateur scalaire pour une source uniforme, $D_1(n) = 2^{-2n}$, où n est le débit fixe par échantillon. La fonction $D_d(R)$, explicitée en section 5.1, donne la performance asymptotique du quantificateur vectoriel pour cette même source.

La gain le plus important entre la TTCQ et la TCQ est obtenu pour un débit de 2 bits par échantillon, et vaut 0.25 dB. Toutefois, à bas débit, il existe des méthodes de quantification qui offrent de meilleures performances que la TTCQ et sont de moindre complexité [133] [134]. Ainsi, le gain à haut débit, bien que plus faible, est plus intéressant. À 8 bits/échantillon, la TTCQ apporte un gain de 0.17dB par rapport à la TCQ. Malheureusement, pour des séquences plus longues, les performances de la TTCQ souffrent du problème de convergence. Notons que l'utilisation de codes de mémoire plus importante (par exemple un code à 16 états de polynômes générateurs $(23, 35)$ en octal), n'a pas permis d'améliorer ces résultats expérimentaux.

débit (bits)	TCQ	TTCQ	quantificateur scalaire	$D_{50}(R)$
1	5.69	**5.90**	6.02	7.28
2	12.39	**12.64**	12.04	13.30
3	18.68	**18.88**	18.06	19.33
4	24.85	**25.03**	24.08	25.34
6	37.00	**37.20**	36.12	37.36
8	49.06	**49.23**	48.16	49.43

TAB. 5.2: Performance en SNR (dB) de la TTCQ par rapport à la TCQ et aux bornes à différents débits pour une séquence de 50 échantillons uniformément distribués.

L'impact de la convergence sur les performances de l'algorithme turbo est illustré dans le tableau 5.3. En utilisant la définition de la convergence de la section 5.6, la distorsion observée sur les vecteurs pour lesquels l'algorithme a convergé après moins de 10^4 itérations est plus petite que la distorsion observée sur les autres séquences. Bien que 10^4 paraisse comme un nombre important d'itérations comparé au codage de canal, elle sont nécessaires pour obtenir un gain en performance. En effet, il y a une différence de 0.33 dB entre la TTCQ avec 100 itérations et la TTCQ avec 10000 itérations pour une séquence de 50 échantillons à 8 bits/échantillons. La complexité totale de la TTCQ est ainsi très élevée comparée à la TCQ pour un même code convolutif et une taille de séquence identique. Puisque l'algorithme SOVA est environ deux fois plus complexe que l'algorithme de Viterbi, pour des performances acceptables, la TTCQ est approximativement 20000 fois plus complexe que la TCQ. Cependant, bien que la

quantizer	TCQ	TTCQ	[convergente/non convergente]
SNR (dB) 2bpp	12.39	12.64	[12.90/12.46]
SNR (dB) 6bpp	37.00	37.20	[37.34/37.10]

TAB. 5.3: TTCQ sur 64 échantillons d'une source uniforme à différents débits, 10^4 itérations.

constante soit plus bien plus grande, l'algorithme reste en temps linéaire par rapport à la longeur de séquence.

Pour la TTCQ, les performances ne s'améliorent pas à mesure que la taille de séquence augmente car le nombre de séquences convergentes décroît rapidement avec la dimension d. Par exemple, le tableau 5.6 montre que pour des blocs de 200 échantillons, l'algorithme turbo converge en moins de 1000 itérations pour seulement 2% des séquences testées, contre 37% pour les séquences de 50 échantillons. En comparaison, les performances de la TCQ augmentent avec la dimension d. En comparant la TCQ à 256 états sur 10000 échantillons à la TTCQ à 4 états sur 50 échantillons, de complexités comparables, la TCQ offre un gain de 0.3 dB par rapport à la TTCQ. Cela signifie que la TTCQ n'a qu'un intérêt pour les applications qui nécessitent une faible latence et ne sont pas limitées par la complexité.

quantification	1 association	2 associations*	4 associations*
quantificateur scalaire	48.16	-	-
borne débit-distorsion	49.43	-	-
TCQ 4 états	49.06	49.11	49.13
TCQ 256 états	49.18	49.20	49.23
TTCQ 4 états, 10^3 itérations	49.16	49.22	49.27
TTCQ 4 états, 10^4 itérations	49.23	49.26	49.29

* Dans le cas où plusieurs associations sont utilisées, il y a un surcoût de $x = 1$ ou $x = 2$ bits par séquence, pour 2 ou 4 associations respectivement. Ce coût supplémentaire est pris compte via une correction des valeurs débit-distorsion, afin d'avoir une comparaison honnête entre les différentes approches. Les valeurs de SNR correspondantes sont calculées par interpolation linéaire entre les valeurs de SNR obtenues pour des débits totaux de $d(n-1)+x$ et $dn+x$.

TAB. 5.4: Influence du nombre d'associations pour une quantification sur 8 bits de séquences uniformément distribuées

Le tableau 5.4 montre comment plusieurs associations influencent les résultats de la TTCQ et la TCQ à 8 bits/échantillons. Dans la TCQ, le choix parmi plusieurs associations permet de minimiser la distorsion introduite par le choix de l'état initial. Le SNR pour la TCQ à plusieurs associations est légèrement meilleur que pour la TCQ simple. Dans la TTCQ, permettre plusieurs associations réduit le nombre d'itérations nécessaire pour obtenir les résultats de la TTCQ à une seule association. Ce choix parmi plusieurs associations augmente la performance de la TTCQ au dépend d'une complexité accrue. Des résultats similaires sont observés pour d'autres débits, avec un gain par rapport à la TCQ d'autant supérieur que le débit augmente. Aussi bien les

résultats du tableau 5.4 que les courbes de la figure 5.10 sont corrigés pour prendre en compte le surcoût d'un ou deux bit(s) nécessaire pour signaler au décodeur quelle association utiliser.

σ	entropie (bits)			SNR (dB)			
	SQ	TCQ	TTCQ	SQ	TCQ	TTCQ	Borne
1	2.10	2.13	2.13	10.79	11.69	11.86	12.32
4	4.05	4.05	4.05	22.83	23.73	23.91	24.37
32	7.04	7.05	7.04	40.89	41.80	41.96	42.43

TAB. 5.5: Entropie et SNR pour une TTCQ uniforme sur 8 bits avec un pas de quantificateur fixé à $\Delta = 1$.

Le tableau 5.5 montre la performance en SNR d'une TTCQ appliquée sur une source gaussienne de variance σ^2. Notons que la quantification uniforme de sources non uniformes n'est pas optimale. Cependant, à haut débit, les dictionnaires contraints en entropie [25] tendent à devenir uniformes. Ainsi, lorsqu'ils sont suivis de codeurs entropiques, les quantificateurs uniformes fournissent de bonnes performances débit-distorsion, même pour les sources non uniformes, comme le montre leur utilisation dans la plupart des systèmes de compression de signaux audio ou vidéo actuels. Ces quantificateurs ont l'avantage de ne pas nécessiter la construction et la transmission d'un dictionnaire au décodeur. Afin de s'assurer que les différentes solutions mènent à des débits similaires lorsqu'elles sont suivies d'un codeur entropique, l'entropie de premier ordre des séquences quantifiées a été mesurée pour les différentes solutions de quantification et figurent dans la première colonne du tableau 5.5.

À haut débit, l'entropie des séquences quantifiées issues de la TTCQ uniforme est proche de l'entropie observée en sortie d'un quantificateur uniforme du même débit. Ainsi l'analyse comparative reste valide lorsque ces quantificateurs sont suivis d'un codeur entropique (le gain relatif entre ces deux quantificateurs est préservé). Les résultats montrent que l'utilisation d'une TCQ ou d'une turbo TCQ uniforme pour quantifier des sources non uniformes permet d'obtenir des gains par rapport à la quantification uniforme dans la mesure où ces quantificateurs sont suivis d'un codeur entropique.

quantificateur	TCQ	TTCQ	2D-TTCVQ
SNR **50 échantillons** (dB)	49.05	49.16	48.99
taux de convergence		36.6%	36.0%
SNR **200 échantillons** (dB)	49.13	48.31	49.17
taux de convergence		2.1%	13.4%

TAB. 5.6: Taux de convergence sur l'ensemble des séquences testées pour 10^3 itérations à 8 bpp pour une source uniforme.

Le tableau 5.6 montre les résultats pour une TTCQ vectorielle. La TTCVQ offre de meilleures performances que la TTCQ pour des séquences plus longues du fait de

meilleures propriétés de convergence. En utilisant une quantification vectorielle bidimensionnelle, elle obtient un gain en performance par rapport à la TCQ pour des séquences de tailles moyennes, bien que ces performances finissent par chuter pour les dimension les plus élevées. La figure 5.10 présente les courbes de SNR par rapport à la longeur des séquences pour divers quantificateurs et une source uniforme à un débit de 8 bits/échantillon. Cette courbe montre le gain apporté par la TTCQ et la QTTCQ par rapport à la TCQ pour des séquences de tailles courtes. Elle montre également qu'un gain pour des séquences plus longues est obtenu avec la TTCVQ. Les performances de la TCQ saturent rapidement à 49.16 dB. Pour des séquences plus courtes, les performances de la QTTCQ sont proches de la borne, mais décroissent rapidement à mesure que la dimension augmente. La TTCVQ obtient de bons résultats pour des séquences de taille moyennes, cependant ses performances décroissent également pour des séquences de plus de 250 échantillons.

5.9 Application au codage de Costa

Nous nous contentons ici de rapporter certains autres travaux pour lesquels les quantificateur TCQ à sortie souple et turbo se sont montrés d'un intérêt particulier. Le codage de Costa [135] est un problème de codage de canal avec information adjacente à l'encodeur, dual du codage de source distribué de Slepian-Wolf [136] et Wiener-Ziv [137]. Ce problème consiste à transmettre un message de l'encodeur au décodeur sur un canal modélisé par $Y = X + S + N$, où S est un bruit blanc gaussien additif connu de l'encodeur uniquement, et N est un bruit blanc gaussien additif inconnu. Ce type de canal est utilisé comme modèle dans les systèmes de communication à canaux multiples ou les systèmes de tatouage. Il a été démontré que la capacité d'un tel canal est donnée par $\frac{1}{2}\log_2(1+\frac{\sigma_X^2}{\sigma_N^2})$, où σ_X^2 et σ_N^2 correspondent respectivement aux puissances du signal et du bruit inconnu. Ce résultat est surprenant car cette capacité correspond également à celle d'un canal à bruit blanc additif gaussien de variance σ_N^2, signifiant que le bruit S ne dégrade théoriquement pas les performances de transmission, bien qu'il soit inconnu du décodeur. Dans un cadre pratique, l'exploitation de l'information adjacente à l'encodeur s'effectue généralement en partitionnant les mots de codes par des techniques de codage de source, comme par exemple la TCQ. En effet, le message à transmettre sélectionne l'une des partitions de la lattice de quantification. L'information adjacente est alors quantifiée sur cette sous-lattice ce qui introduit une distorsion de quantification minime (Fig. 5.11). Le message est retrouvé au décodeur en recherchant le mot de code dans la lattice initiale le plus proche de l'observation au moyen d'un décodeur de canal. L'utilisation d'une TCQ souple au lieu d'une TCQ classique dans un schéma conjoint source-canal itératif de codage de Costa a été proposée dans [138], s'approchant de la capacité en offrant le même taux d'erreur binaire que les techniques antérieures pour un SNR inférieur de 0.27 dB. L'utilisation d'une TCQ souple dans ce schéma a en effet permis d'utiliser le même code convolutif poinçonné lors de la quantification que le code composant la TTCM utilisée pour la transmission sur le canal. Ceci est rendu possible par la prise en compte du poinçonnage en tant qu'information a priori lors de

la quantification.

Le schéma de Costa a été utilisé entre autre dans le cadre de la stéganographie. Cette problématique consiste à transmettre un message caché (binaire) à un destinataire par l'intermédiaire d'un média, telle une image ou un son, sans qu'il soit détectable par un tiers. Toutes les techniques de tatouage et de stéganographie considèrent le média comme un bruit sur le message à transmettre. Parmi ces techniques basées sur le schéma de Costa, dont la spécificité est de tirer parti de la connaissance du média S à l'insertion, certaines consistent à insérer le message dans le média en quantifiant le média moyennant un dictionnaire vectoriel dépendant du message. La précision du quantificateur règle alors la puissance d'insertion du message dans le média et donc la distorsion apportée à celui-ci, ainsi que la robustesse du message aux dégradations ultérieures du média. Afin que la distribution du média quantifié ne trahisse pas le fait qu'il a été altéré, un bruit est ajouté au média avant quantification, puis soustrait après quantification (lissage). Ce bruit est issu d'un générateur pseudo-aléatoire dont la graine est la clé secrète que l'envoyeur doit avoir fournie au destinataire. Dans le cadre de la quantification TCQ, le message binaire sélectionne un chemin dans le treillis du code convolutif utilisé par le quantificateur. La séquence de dictionnaires de quantification scalaire obtenue est ensuite utilisée pour quantifier chaque échantillon du média. Au décodeur, moyennant la clé secrète, le même bruit qu'à l'encodeur est ajouté à nouveau au média. Ce dernier est finalement quantifié à l'aide d'une TCQ et le chemin choisi par l'algorithme de Viterbi correspond au message transmis (Fig. 5.12). La technique de stéganographie de média audio proposée dans [139]* utilise un quantificateur turbo TCQ permettant d'obtenir un gain d'environ 5.5dB par rapport à un quantificateur scalaire en termes de puissance d'insertion par rapport au bruit pour un taux d'erreur binaire de 10^{-5}.

5.10 Conclusion

Les codes convolutifs et les principes de partionnement d'ensembles ont été appliqués avec succès à la quantification de sources, menant à la quantification codée par treillis. Pour minimiser l'erreur quadratique de reconstruction, le code de partionnement doit être tel qu'il maximise la distance entre les séquences quantifiées admissibles. De la même manière que les turbo codes ont été utilisés en TTCM pour améliorer les propriétés de distances entre les séquences modulées, nous avons considéré ici l'utilisation de turbo codes pour le partionnement afin d'augmenter la distance entre les séquences quantifiées. Nous avons tout d'abord conçu une TCQ à sortie souple afin de l'utiliser comme composante de la turbo TCQ. Une structure turbo TCQ parallèle a été décrite ainsi que l'algorithme itératif de quantification associé. Le comportement en convergence de cet algorithme a été analysé en effectuant un parallèle avec la TTCM. Du fait de différences fondamentales dans la distribution du bruit, le nombre d'itérations nécessaire pour obtenir la convergence de l'algorithme est très grand (impliquant une complexité importante), et la convergence de l'algorithme peut même échouer, en particulier quand la séquence est trop longue. Toutefois, la TTCQ apporte un gain en performance par

*basée sur notre implémentation (`http ://www.irisa.fr/temics/Equipe/Chappelier/turbo_tcq.tar.gz`)

rapport à la TCQ pour des séquences courtes à moyennes, pour un coût important en complexité. Notons que, bien que la complexité du quantificateur itératif soit très grande, le processus de déquantification reste très simple. La TTCQ a été étendue au cas vectoriel de la TTCVQ, apportant des améliorations par rapport à la TCQ pour des séquences de tailles moyennes. Accroître la dimension des sous-dictionnaires de quantification vectorielle utilisés dans la TTCVQ devrait permettre d'améliorer encore les performances pour des séquences plus longues.

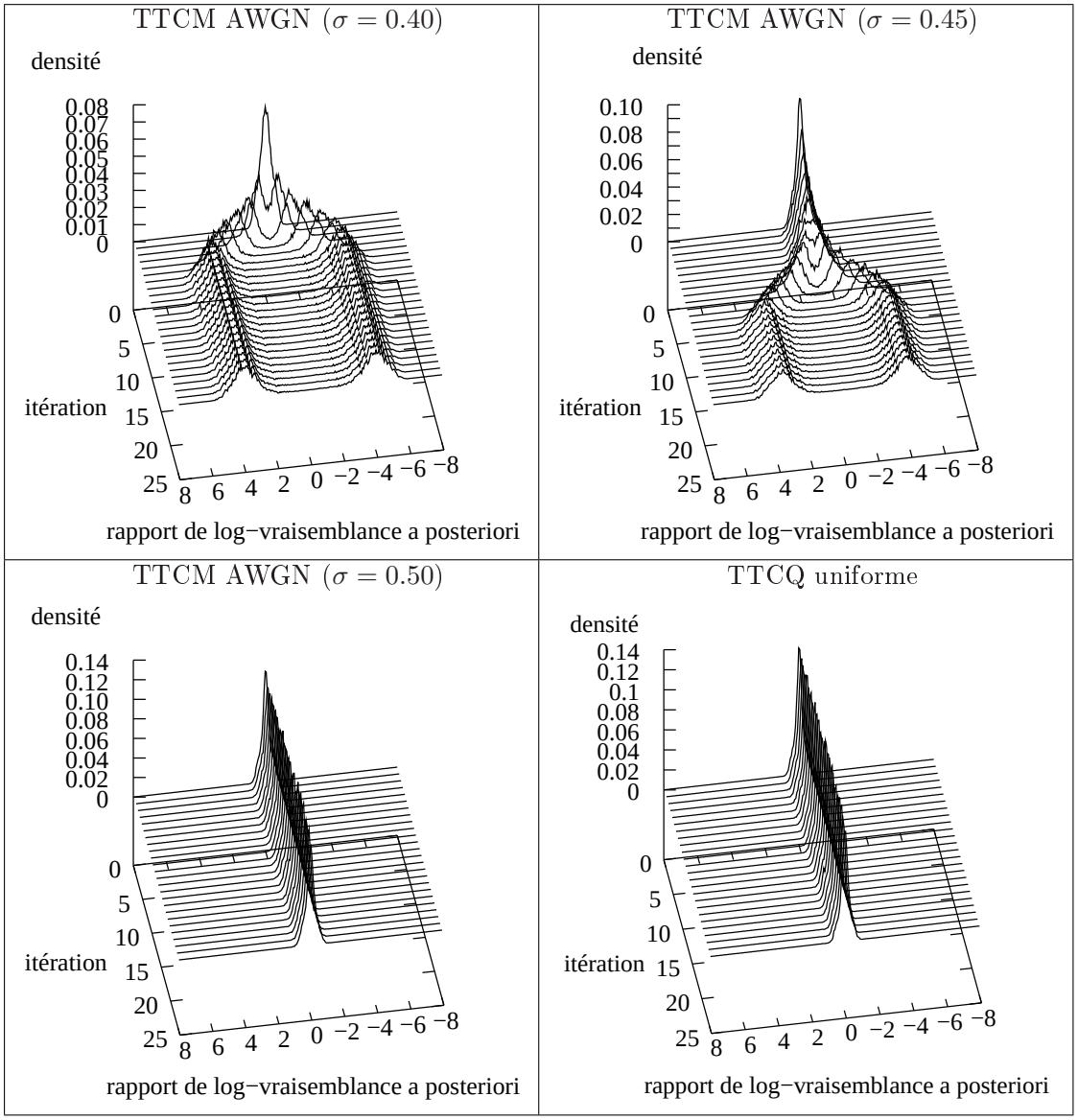

FIG. 5.8: Distribution des rapports de log-vraisemblance a posteriori pour différents niveaux de bruit gaussien dans le cas de TTCM et pour la TTCQ uniforme. Lorsque le bruit est limité, l'algorithme converge et deux modes apparaissent clairement, correspondants aux deux états possibles des bits de la séquence [haut gauche]. L'algorithme converge après un plus grand nombre d'itérations lorsque la puissance de ce bruit blanc additif est proche de la limite de Shannon [haut droit]. Lorsque la puissance du bruit est trop proche de la limite ou la dépasse, les métriques a posteriori restent centrées ; les bits en sortie sont donc aléatoires et la séquence décodée est fausse [bas gauche]. De même, pour la TTCQ uniforme, les métriques restent généralement centrées après un grand nombre d'itérations [bas droit].

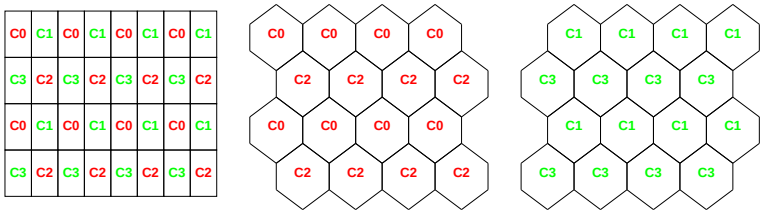

FIG. 5.9: Partitionnement du dictionnaire uniforme bidimensionnel $\mathcal{C}$ en deux classes d'équivalences $C_0 \bigcup C_2$ et $C_1 \bigcup C_3$ formant chacune une lattice hexagonale régulière.

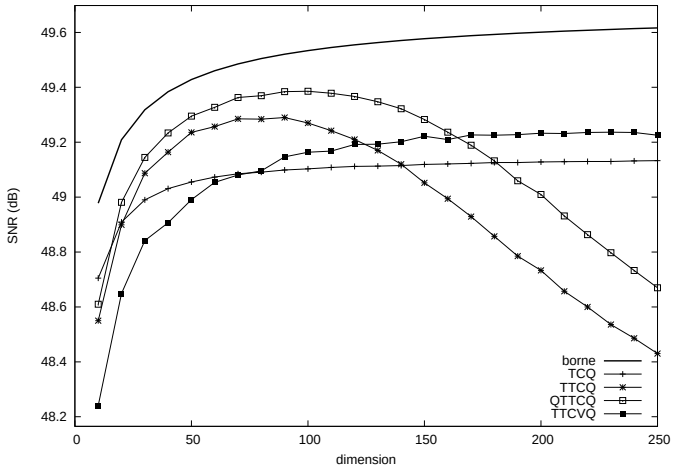

FIG. 5.10: Distorsion en fonction de la longueur de séquence pour différents quantificateurs sur 256 niveaux.

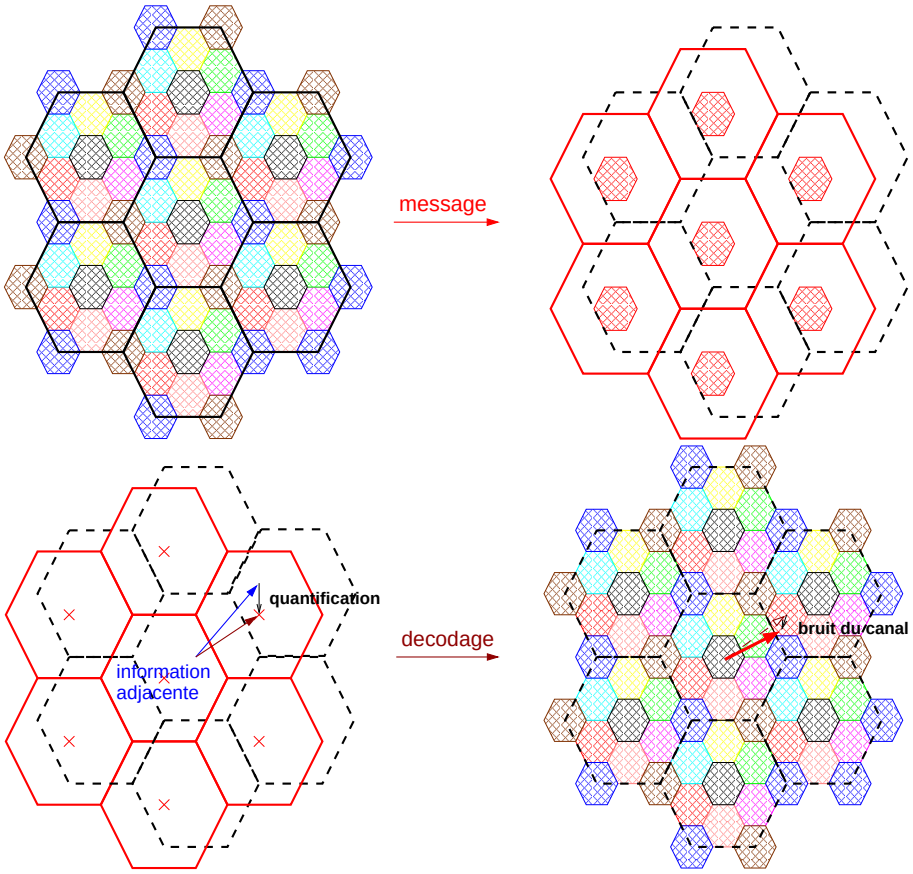

FIG. 5.11: Principe du codage de Costa par quantification et codage canal. L'ensemble des mots de codes d'un dictionnaire de source formant une lattice est partitionné en sous-lattices [haut gauche]. Le message à transmettre selectionne l'une de ces sous-lattices [haut droit] pour effectuer la quantification de l'information adjacente [bas gauche]. Le mot de code correspondant est ensuite transmis sur le canal après protection par un code de canal. Un décodage de l'observation bruitée permet de retrouver le mot de code original. La sous-lattice à laquelle il appartient identifie le message transmis [droite].

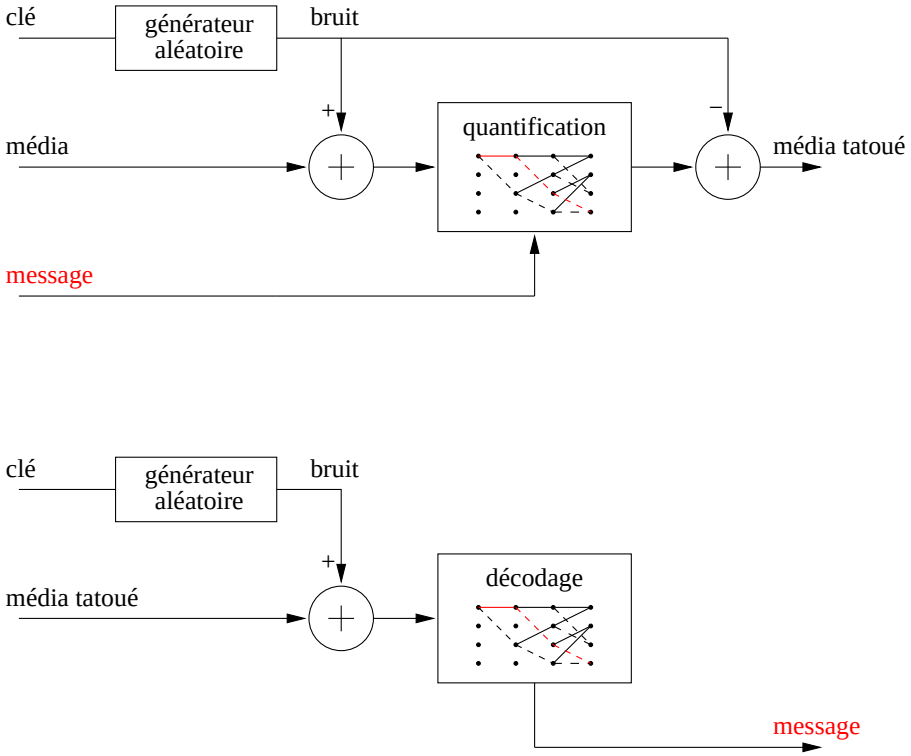

FIG. 5.12: Schéma de principe de la stéganographie par quantification TCQ. L'insertion s'effectue en quantifiant le média moyennant un chemin donné par le message à transmettre, après lissage par un bruit généré par la clé secrète [haut]. L'extraction du message s'obtient en appliquant une quantification TCQ sur le média tatoué, sélectionnant le chemin correspondant au message [bas].

Conclusion

Au cours de cette thèse, nous nous sommes intéressés aux différentes étapes d'une chaîne de compression d'images numériques, en proposant de nouvelles techniques de transformées et de quantification. Nous avons également présenté l'intérêt de ces techniques dans d'autres domaines du traitement du signal, liés aux thématiques du codage vidéo, du débruitage, ou encore de la stéganographie. Bien que centrée sur les techniques de transformées orientées, cette étude a montré l'importance de considérer l'ensemble de la chaîne de codage pour évaluer les performances des systèmes de compression. En effet, l'impact de la redondance et de la corrélation résiduelle sur les performances des codeurs de sous-bandes rend difficile l'évaluation directe des diverses transformées par simple comparaison de leurs performances d'approximation non-linéaire.

Dans ce cadre nous avons tout d'abord proposé d'adapter la transformée en contourlettes au codage d'images et de vidéos. Cette transformée étant redondante, nous l'avons combinée avec une transformée en ondelettes séparables afin de pouvoir contrôler la redondance totale apportée. Dans cette approche, les informations de contour de hautes fréquences sont codées par contourlettes, tandis que les basses fréquences sont codées par ondelettes séparables. Nous avons également appliqué une technique de projection sur ensembles convexes pour optimiser les performances de cette transformée redondante en présence de bruit de quantification. Le couplage de cette transformée au codeur EZBC, nous a permis d'en évaluer les performances par rapport à la transformée en ondelettes séparables, et d'observer un gain de codage à très bas débit.

Nous avons également proposé une nouvelle transformée orientée adaptative afin de s'affranchir des problèmes de redondance à haut débit. Cette technique repose sur une structure d'échantillonnage multirésolution quinconce sur laquelle est définie une carte d'orientation. Cette carte permet d'aligner les pas lifting d'une ondelette dans la direction des contours. Le passage de l'ondelette unidimensionnelle à l'ondelette orientée a tout d'abord été présenté. Nous avons ensuite proposé plusieurs méthodes de recherche de la carte d'orientation en fonction de l'application. Dans le cadre du codage d'images, nous avons proposé de coder la carte d'orientation à l'aide de quad-tree et d'une optimisation débit-distorsion entre le débit alloué à la carte et le débit alloué aux coefficients d'ondelettes. Nous avons ensuite évalué les performances de cette nouvelle transformée par rapport à la transformée en ondelettes séparables en les combinant avec le codeur EBCOT. Afin d'évaluer la pertinence des contextes utilisés pour le codage en sous-bande, nous avons également comparé l'information mutuelle résiduelle entre les coefficients d'ondelettes orientées voisins par rapport aux résultats observés sur les co-

efficients d'ondelettes séparables. Cette étude nous a permis de conclure que bien qu'un gain plus faible soit attendu de l'exploitation de cette information par les codeur de sous-bandes, une dépendance résiduelle limitée existe encore entre les coefficients d'ondelettes orientées appartenant à une même sous-bande. Enfin, nous avons appliqué cette transformée au problème du débruitage d'images afin de la comparer à la transformée en ondelettes séparables. En modélisant la carte d'orientation par un champ de Markov, et en utilisant une technique simple de seuillage des coefficients d'ondelettes, nous avons obtenu des performances comparables à l'état de l'art et supérieure d'environ 0.5 dB aux performances de la transformée en ondelettes séparables.

Finalement, dans le but d'améliorer les performances des quantificateurs utilisés en compression, nous nous sommes intéressés à l'utilisation de codes turbo pour la quantification TCQ. En s'appuyant sur la dualité source-canal des systèmes de TCM et de TCQ, nous nous sommes inspirés de la modulation turbo TCM pour proposer un système de quantification turbo TCQ. Dans ce système, le code convolutif utilisé par le quantificateur TCQ a été remplacé par un code turbo, menant à un algorithme itératif de quantification. Bien que ces systèmes de quantification et de modulation soient très similaires, nous avons montré que la distribution des données sur lesquels ils s'appliquent change totalement le comportement itératif de l'algorithme turbo sur lequel ils reposent. En se basant sur des arguments de géométrie et de théorie de l'information, nous avons montré que contrairement au cas de la modulation, l'algorithme turbo ne peut converger vers un point fixe pour la quantification de séquences longues. Nous avons toutefois proposé différentes modifications du quantificateur turbo TCQ pour en améliorer les performances sur les séquences courtes, permettant de s'approcher à 0.2 dB des performances optimales de quantification vectorielle pour des dimensions de l'ordre de la cinquantaine d'échantillons.

L'étude menée dans le cadre des contourlettes a montré qu'il est difficile d'obtenir de bonnes performances en compression avec des transformées redondantes à haut débit. Toutefois, le gain observé à bas débit et la qualité visuelle des contours restitués montre que ces approches géométriques pourraient apporter une amélioration aux systèmes de compression actuels. Un second problème de ces approches par bancs de filtres orientés réside dans le fait qu'elles associent une orientation fixe à chaque position dans l'image. Ainsi, plus le nombre d'orientations d'analyse augmente, moins les contours sont localisés précisément par la transformée. En proposant une approche adaptative non redondante, nous avons tenté de séparer le problème du codage de la géométrie de l'image de celui du codage de sa texture. Pour simplifier l'approche et limiter les problèmes d'orthogonalité, nous nous sommes restreints à un choix d'orientations binaire en chaque point. Ceci nous a permis de définir quatre orientations d'analyse en fonction de l'échelle considérée. Vue de manière globale, cette transformée réalise un compromis entre les approches par bancs de filtres fixes et les approches purement géométriques spécifiant explicitement l'orientation de chaque point de l'image. En effet, à chaque position dans l'image est associée une paire de fonctions de base de même échelle et d'orientations orthogonales. La géométrie est codée en choisissant une fonction particulière adaptée à l'orientation locale. Le lien avec la structure statique des bancs de filtres réside dans le fait que ces deux orientations sont soit verticales et horizontales, soit diagonales et an-

tidiagonales en fonction de la position du coefficient associé dans l'image. Nous pensons qu'il serait intéressant de considérer ce point de vue pour étendre l'approche à un plus grand choix d'orientations, et rechercher le meilleur compromis entre la précision en localisation et en orientation, tout en restant à échantillonnage critique. Il serait également utile de considérer d'autres structures de sous-échantillonnage que les structures quinconces ou séparables, afin de déterminer celles qui réalisent le meilleur compromis entre précision en position et en orientation pour les images naturelles. De plus, les filtres utilisés dans cette approche sont identiques à chaque échelle. Il serait intéressant de concevoir des filtres adaptés à chaque échelle afin de permettre une concentration maximale de l'énergie dans les basses fréquences.

Les codeurs de sous-bandes utilisés dans cette thèse n'ont pas été adaptés aux transformées considérées. Il serait possible de calibrer les contextes utilisés dans ces codeurs adaptatifs aux transformées orientées, bien que le gain de codage à en attendre soit limité. Il serait également possible d'utiliser l'information d'orientation comme contexte de codage. Au niveau de la quantification, l'application de la turbo TCQ à la quantification de sous-blocs de coefficients pourrait apporter un léger gain si la complexité n'est pas problématique. Dans ce cadre, il serait possible de l'étendre de la même manière que la TCQ, en modifiant les métriques utilisée pour tenir compte du débit et en effectuant un partitionnement identique à celui proposé dans la TCQ universelle. Il serait également intéressant de considérer la possibilité de réaliser un quantificateur TCQ basé sur des codes LDPC, tout en s'intéressant aux algorithmes d'inférence dans les graphes de facteurs pour en étudier la convergence dans le cadre de la quantification.

Preuves

Chapitre 1 : Cadre théorique

Proposition 1. *Une lattice possède une structure de groupe additif commutatif.*

Démonstration. Soit $\mathsf{x},\mathsf{y},\mathsf{z} \in \mathsf{A}\mathbb{Z}^d, \exists \mathsf{i},\mathsf{j},\mathsf{k} \in \mathbb{Z}^d, \mathsf{x} = \mathsf{A}\mathsf{i}, \mathsf{y} = \mathsf{A}\mathsf{j}, \mathsf{z} = \mathsf{A}\mathsf{k}$

- stabilité : $\mathsf{x}+\mathsf{y} = \mathsf{A}\mathsf{i}+\mathsf{A}\mathsf{j} = \mathsf{A}(\mathsf{i}+\mathsf{j}) \in \mathsf{A}\mathbb{Z}^d$
- associativité : $(\mathsf{x}+\mathsf{y})+\mathsf{z} = \mathsf{A}((\mathsf{i}+\mathsf{j})+\mathsf{k}) = \mathsf{A}(\mathsf{i}+(\mathsf{j}+\mathsf{k})) = \mathsf{x}+(\mathsf{y}+\mathsf{z})$
- neutre : $\mathsf{A}0 = 0 \in \mathsf{A}\mathbb{Z}^d, \mathsf{x}+0 = \mathsf{A}(\mathsf{i}+0) = \mathsf{A}\mathsf{i} = \mathsf{x}$
- inverse : $\mathsf{A}(-\mathsf{i}) \in \mathsf{A}\mathbb{Z}^d$ et $-\mathsf{x}+\mathsf{x} = \mathsf{A}(-\mathsf{i}+\mathsf{i}) = 0$
- commutativité : $\mathsf{x}+\mathsf{y} = \mathsf{A}(\mathsf{i}+\mathsf{j}) = \mathsf{A}(\mathsf{j}+\mathsf{i}) = \mathsf{y}+\mathsf{x}$

□

Proposition 2. $\forall \mathsf{A} \in \boldsymbol{GL}_d(\mathbb{R}), \mathsf{A}\mathbb{Z}^d$ *est isomorphe à* $\mathbb{Z}^d$.

Démonstration. L'application $H : \begin{array}{ccc} \mathbb{Z}^d & \to & \mathsf{A}\mathbb{Z}^d \\ \mathsf{z} & \mapsto & \mathsf{A}\mathsf{z} \end{array}$ est linéaire et bijective, il s'agit donc d'un isomorphisme. □

Remarque 1. *Le volume de la cellule élémentaire* $\mathsf{A}[0,1[^d$ *vaut* $|\det(\mathsf{A})|$.

Démonstration. Soit l'application linéaire $\chi(\mathsf{t}) = \mathsf{A}\mathsf{t}$. Cette application définit un changement de variable dont la matrice jacobienne définie par $\frac{d\chi}{dt}$ vaut A. Par application du théorème de changement de variable on a :

$$\int_{\mathsf{A}[0,1[^d} d\mathsf{t} = \int_{[0,1[^d} |\det(\mathsf{A})| d\mathsf{t} = |\det(\mathsf{A})|$$

□

Proposition 3. ***(Division euclidienne)*** *Soit* $\mathsf{M} \in \mathcal{M}_d(\mathbb{Z}), \det(\mathsf{M}) \neq 0$ *une matrice carrée entière de déterminant non nul. Soit l'ensemble* $\mathcal{R}^d_{\mathsf{M}} = \mathbb{Z}^d \cap \mathsf{M}[0,1[^d$. *Alors*

$$\forall \mathsf{z} \in \mathbb{Z}^d, \exists ! \mathsf{q} \in \mathbb{Z}^d, \mathsf{r} \in R^d_{\mathsf{M}}, \mathsf{z} = \mathsf{M}\mathsf{q} + \mathsf{r}$$

Démonstration. Comme $\det(\mathsf{M}) \neq 0$, $\mathsf{M} \in \mathbf{GL}_d(\mathbb{Q})$. Soit $\mathsf{z} \in \mathbb{Z}^d, \exists! \, \mathsf{y} = M^{-1}\mathsf{z} \in \mathbb{Q}^d$. Alors, par division euclidienne dans $\mathbb{Q}$, chaque composante de y s'écrit de manière unique $\mathsf{y}_i = \mathsf{q}_i + \mathsf{r}_i$ avec $\mathsf{q}_\mathsf{i} \in \mathbb{Z}$ partie entière et $\mathsf{r}_\mathsf{i} \in [0,1[$ partie fractionnaire de y_i. Ainsi

$$\exists! \mathsf{q} \in \mathbb{Z}^d, \mathsf{b} \in [0,1[^d, \mathsf{y} = \mathsf{q} + \mathsf{b}.$$

Or $\mathsf{z} = \mathsf{My} = \mathsf{M}(\mathsf{q} + \mathsf{b})$. Puisque $\mathsf{z} \in \mathbb{Z}^d$ et $\mathsf{Mq} \in \mathbb{Z}^d$ on a $\mathsf{Mb} = \mathsf{z} - \mathsf{Mq} \in \mathbb{Z}^d$. Or $\mathsf{Mb} \in \mathsf{M}[0,1[^d$, donc $\mathsf{r} = \mathsf{Mb} \in \mathcal{R}^d_\mathsf{M}$. □

Corrolaire 1. *$\mathcal{R}^d_\mathsf{M}$ est un ensemble de représentants de $\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d$. On notera par la suite de manière unique $[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d] = \mathcal{R}^d_\mathsf{M}$.*

Démonstration. D'après la proposition précédente, $\mathbb{Z}^d \subseteq \mathsf{M}\mathbb{Z}^d + \mathcal{R}^d_\mathsf{M}$. La réciproque est triviale entraînant $\mathbb{Z}^d = \mathsf{M}\mathbb{Z}^d + \mathcal{R}^d_\mathsf{M}$. Supposons qu'il y ait deux représentants différents u et v d'une même classe dans $\mathcal{R}^d_\mathsf{M}$. Alors $\exists k \in \mathbb{Z}^d \backslash \{0\}, u = v + \mathsf{M}k$ donc $u \notin M[0,1[^d$: contradiction. □

Lemme 1.

$$\forall t \in \mathbb{Z}^d, f(\mathsf{t}) = \int\limits_{\mathsf{M}[0,1[^d} e^{2i\pi(\mathsf{M}^{-1}\mathsf{t})^\top \mathsf{x}} d\mathsf{x} = \begin{cases} |\det(\mathsf{M})| & si \ \mathsf{t} = 0 \\ 0 & sinon. \end{cases}$$

Démonstration. Soit $\mathsf{t} \in \mathbb{Z}^d$, par changement de variable $\mathsf{x} \mapsto \mathsf{M}^\top \mathsf{y}$ on a :

$$\int\limits_{M[0,1[^d} e^{2i\pi(\mathsf{M}^{-1}\mathsf{t})^\top \mathsf{x}} d\mathsf{x} = \int\limits_{[0,1[^d} e^{2i\pi \mathsf{t}^\top \mathsf{y}} |\det(\mathsf{M}^\top)| d\mathsf{y} = |\det(\mathsf{M})| \prod_{k=0}^{d-1} \int_0^1 e^{2i\pi t_k y_k} dy_k$$

Posons

$$c_{t_k} = \int_0^1 e^{2i\pi t_k y_k} dy_k.$$

Alors, soit $t_k = 0$ et $c_{t_k} = c_0 = 1$, soit $t_k \neq 0$ et

$$c_{t_k} = \frac{1}{2i\pi t_k}(e^{2i\pi t_k} - 1).$$

Comme $t_k \in \mathbb{Z}$, on a

$$c_{t_k} = \begin{cases} 1 & si \ t_k = 0 \\ 0 & sinon. \end{cases}$$

Ainsi,

$$|\det(\mathsf{M})| \prod_{k=0}^{d-1} c_{t_k} = \begin{cases} |\det(\mathsf{M})| & si \ \mathsf{t} = 0 \\ 0 & sinon. \end{cases}$$

Donc,

$$f(\mathsf{t}) = \begin{cases} |\det(\mathsf{M})| & si\ \mathsf{t} = 0 \\ 0 & sinon. \end{cases}$$

□

Proposition 4. ***(Somme des racines de l'unité)*** *Soit* $\mathsf{M} \in \mathcal{M}_d(\mathbb{Z}), \det(\mathsf{M}) \neq 0$ *une matrice carrée entière de déterminant non nul.*

$$\sum_{\mathsf{x} \in \mathcal{R}_{\mathsf{M}}^d} e^{2i\pi (M^{-1}\mathsf{t})^\top \mathsf{x}} = \begin{cases} |\det M| & si\ \mathsf{t} \in \mathsf{M}\mathbb{Z}^d \\ 0 & sinon. \end{cases}$$

Démonstration. Posons $h(t) = \sum_{\mathsf{k} \in \mathsf{M}\mathbb{Z}^d} f(t-k)$. D'après le lemme,

$$\forall t \in \mathbb{Z}^d, h(t) = \begin{cases} |\det M| & si\ \mathsf{t} \in \mathsf{M}\mathbb{Z}^d \\ 0 & sinon. \end{cases}$$

Or

$$\begin{aligned} &\sum_{\mathsf{k} \in \mathsf{M}\mathbb{Z}^d} \int_{\mathsf{M}[0,1[^d} e^{2i\pi (\mathsf{M}^{-1}(\mathsf{t}-\mathsf{k}))^\top \mathsf{x}} d\mathsf{x} \\ &= \sum_{\mathsf{a} \in \mathbb{Z}^d} \int_{\mathsf{M}[0,1[^d} e^{2i\pi (\mathsf{M}^{-1}\mathsf{t})^\top \mathsf{x}} e^{-2i\pi \mathsf{a}^\top \mathsf{x}} d\mathsf{x} \\ &= \int_{\mathsf{M}[0,1[^d} e^{2i\pi (\mathsf{M}^{-1}\mathsf{t})^\top \mathsf{x}} \sum_{\mathsf{a} \in \mathbb{Z}^d} e^{-2i\pi \mathsf{a}^\top \mathsf{x}} d\mathsf{x} \\ &= \int_{\mathsf{M}[0,1[^d} e^{2i\pi (\mathsf{M}^{-1}\mathsf{t})^\top \mathsf{x}} \delta_{\mathbb{Z}^d}(\mathsf{x}) d\mathsf{x} \\ &= \sum_{\mathsf{x} \in \mathcal{R}_{\mathsf{M}}^d} e^{2i\pi (\mathsf{M}^{-1}\mathsf{t})^\top \mathsf{x}} \end{aligned}$$

□

Corrolaire 2. *Le cardinal de* $[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]$ *vaut* $|\det(\mathsf{M})|$

Démonstration. Pour $\mathsf{t} = 0$ on obtient $|\mathcal{R}_{\mathsf{M}}^d| = \sum_{\mathsf{n} \in \mathcal{R}_{\mathsf{M}}} 1 = |\det(\mathsf{M})|$. □

Proposition 5. *(sur-échantillonnage) La transformée de Fourier d'un signal sur-échantillonné par* M *s'exprime en fonction de la transformée de Fourier du signal original par :*

$$X_{\uparrow \mathsf{M}}(\mathsf{f}) = X(\mathsf{M}^\top \mathsf{f}) \tag{8}$$

Démonstration.

$$\begin{aligned} X_{\uparrow \mathsf{M}}(\mathsf{z}) &= \sum_{\mathsf{n}\in\mathbb{Z}^d} x_{\uparrow \mathsf{M}}[\mathsf{n}] e^{-2i\pi \mathsf{f}^\top \mathsf{n}} = \sum_{\mathsf{n}\in \mathsf{M}\mathbb{Z}^d} x[\mathsf{M}^{-1}\mathsf{n}] e^{-2i\pi \mathsf{f}^\top \mathsf{n}} = \sum_{\mathsf{n}\in\mathbb{Z}^d} x[\mathsf{n}] e^{-2i\pi \mathsf{f}^\top \mathsf{M}\mathsf{n}} \\ &= \sum_{\mathsf{n}\in\mathbb{Z}^d} x[\mathsf{n}] e^{-2i\pi (\mathsf{M}^\top \mathsf{f})^\top \mathsf{n}} = X(\mathsf{M}^\top \mathsf{f}) \end{aligned}$$

□

Proposition 6. *(sous-échantillonnage) La transformée de Fourier d'un signal sous-échantillonné par* M *s'exprime en fonction de la transformée de Fourier du signal original par :*

$$X_{\downarrow \mathsf{M}}(\mathsf{f}) = \frac{1}{|\det(\mathsf{M})|} \sum_{\mathsf{k}\in[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]} X(\mathsf{M}^{-\top}(\mathsf{f}-\mathsf{k})) \tag{9}$$

Démonstration.

$$\begin{aligned} X_{\downarrow \mathsf{M}}(\mathsf{z}) &= \sum_{\mathsf{n}\in\mathbb{Z}^d} x_{\downarrow \mathsf{M}}[\mathsf{n}] e^{-2i\pi \mathsf{f}^\top \mathsf{n}} = \sum_{\mathsf{n}\in\mathbb{Z}^d} x[\mathsf{M}\mathsf{n}] e^{-2i\pi \mathsf{f}^\top \mathsf{n}} = \sum_{\mathsf{n}\in \mathsf{M}\mathbb{Z}^d} x[\mathsf{n}] e^{-2i\pi \mathsf{f}^\top \mathsf{M}^{-1}\mathsf{n}} \\ &= \sum_{\mathsf{n}\in\mathbb{Z}^d} \left(\frac{1}{|\det(\mathsf{M})|} \sum_{\mathsf{k}\in[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]} e^{2i\pi (M^{-1}\mathsf{n})^\top \mathsf{k}} \right) x[\mathsf{n}] e^{-2i\pi \mathsf{f}^\top \mathsf{M}^{-1}\mathsf{n}} \\ &= \frac{1}{|\det(\mathsf{M})|} \sum_{\mathsf{k}\in[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]} \sum_{\mathsf{n}\in\mathbb{Z}^d} e^{-2i\pi (\mathsf{f}^\top \mathsf{M}^{-1}\mathsf{n} - (M^{-1}\mathsf{n})^\top \mathsf{k})} x[\mathsf{n}] \\ &= \frac{1}{|\det(\mathsf{M})|} \sum_{\mathsf{k}\in[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]} \sum_{\mathsf{n}\in\mathbb{Z}^d} e^{-2i\pi ((\mathsf{M}^{-\top}\mathsf{f})^\top \mathsf{n} - (M^{-\top}\mathsf{k})^\top \mathsf{n})} x[\mathsf{n}] \\ &= \frac{1}{|\det(\mathsf{M})|} \sum_{\mathsf{k}\in[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]} X(\mathsf{M}^{-\top}\mathsf{f} - \mathsf{M}^{-\top}\mathsf{k}) \end{aligned}$$

□

Corrolaire 3. *(repliement de spectre) Le sous-échantillonnage suivi du sur-échantillonnage d'un signal* x *entraîne une réplication de son spectre fréquentiel selon la lattice issue de* $\mathsf{M}^{-\top}$*, appelée lattice réciproque de* $\mathsf{M}\mathbb{Z}^d$ *:*

$$X_{\downarrow \mathsf{M}\uparrow \mathsf{M}}(\mathsf{f}) = \frac{1}{|\det(\mathsf{M})|} \sum_{\mathsf{k}\in[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]} X(\mathsf{f} - \mathsf{M}^{-\top}\mathsf{k})$$

Démonstration.

$$X_{\downarrow \mathsf{M}\uparrow \mathsf{M}}(\mathsf{f}) = X_{\downarrow \mathsf{M}}(\mathsf{M}^\top \mathsf{f}) = \frac{1}{|\det(\mathsf{M})|} \sum_{\mathsf{k}\in[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]} X(\mathsf{M}^{-\top}(\mathsf{M}^\top \mathsf{f} - \mathsf{k}))$$

□

Théorème 1. *Pour obtenir la reconstruction parfaite $\hat{X} = X$ il faut et il suffit que*

$$\begin{cases} H_0(\mathsf{f})G_0(\mathsf{f}) + H_1(\mathsf{f})G_1(\mathsf{f}) = 2 \\ H_0(\mathsf{f} - \mathsf{M}^{-\top}\mathsf{j})G_0(\mathsf{f}) + H_1(\mathsf{f} - \mathsf{M}^{-\top}\mathsf{j})G_1(\mathsf{f}) = 0 \end{cases}. \tag{10}$$

en notant j *l'unique élément non nul de* $[\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]$.

Démonstration. La relation liant le signal Y_L en sortie d'un canal du banc de filtre au signal d'origine X (Fig. 1.6) est donnée par :

$$\begin{aligned} Y_L(\mathsf{f}) &= [H_0 \cdot X]_{\downarrow \mathsf{M}}(\mathsf{f}) \\ &= \frac{1}{|\det(\mathsf{M}|)} \sum_{\mathsf{k} \in [\mathbb{Z}^d/\mathsf{M}\mathbb{Z}^d]} H_0(\mathsf{M}^{-\top}(\mathsf{f} - \mathsf{k}))X(\mathsf{M}^{-\top}(\mathsf{f} - \mathsf{k})) \\ &= \frac{1}{2}(H_0(\mathsf{M}^{-\top}\mathsf{f})X(\mathsf{M}^{-\top}\mathsf{f}) + H_0(\mathsf{M}^{-\top}(\mathsf{f} - \mathsf{j}))X(\mathsf{M}^{-\top}(\mathsf{f} - \mathsf{j})) \end{aligned}$$

Une relation similaire existe entre Y_H et X. Ainsi, on obtient le signal reconstruit :

$$\begin{aligned} \hat{X}(\mathsf{f}) &= Y_{L\uparrow \mathsf{M}}(\mathsf{f})G_0(\mathsf{f}) + Y_{H\uparrow \mathsf{M}}(\mathsf{f})G_1(\mathsf{f}) = Y_L(M^{\top}\mathsf{f})G_0(\mathsf{f}) + Y_H(\mathsf{M}^{\top}\mathsf{f})G_1(\mathsf{f}) \\ &= \underbrace{\frac{1}{2}(H_0(\mathsf{f})G_0(\mathsf{f}) + H_1(\mathsf{f})G_1(\mathsf{f}))X(\mathsf{f})}_{signal} + \\ &\qquad \underbrace{\frac{1}{2}(H_0(\mathsf{f} - \mathsf{M}^{-\top}\mathsf{j})G_0(\mathsf{f}) + H_1(\mathsf{f} - \mathsf{M}^{-\top}\mathsf{j})G_1(\mathsf{f}))X(\mathsf{f} - \mathsf{M}^{-\top}\mathsf{j})}_{aliasing} \end{aligned}$$

Les conditions énoncées par le théorème permettent donc d'annuler la composante d'aliasing $X(\mathsf{f} - \mathsf{M}^{-\top}\mathsf{j})$ et de récupérer le signal $X(\mathsf{f})$. □

Chapitre 5 : Turbo TCQ

Nous calculons ici la probabilité que l'algorithme turbo converge vers un point fixe dans le cas de la TTCQ et de la TTCM. Il est supposé que l'algorithme converge si et seulement si le vecteur d'entrée sur lequel l'algorithme est appliqué est suffisamment proche de la solution dans le dictionnaire $\mathcal{C}$. Ainsi, pour un rayon de convergence r_C, l'algorithme est supposé converger vers $\hat{\mathsf{x}}$ si et seulement si le vecteur d'entrée x vérifie :

$$\|\mathsf{X} - \hat{\mathsf{X}}\| < r_C.$$

Cette hypothèse a été confirmée par les résultats expérimentaux. Nous nous plaçons également à haut débit, auquel cas r_C est petit.

Pour une TTCQ sur n-bit d'une source uniforme distribuée sur $[0, a]$, l'algorithme est lancé sur le vecteur source X_s de taille d, modélisé par

$$\mathsf{X}_s = \hat{\mathsf{X}} + \mathsf{N}_s,$$

où $\hat{\mathsf{X}} \in \mathcal{C}$ est le vecteur quantifié, et N_s le bruit de quantification décorrélé mais dépendant. Ce bruit est constant sur la cellule de Voronoï et donc de densité de probabilité égale à $\frac{2^{nd}}{a^d}$ à l'intérieur de la cellule et 0 en dehors.

Pour une TTCM 2^n-aire sur un canal à bruit blanc additif gaussien, l'algorithme est appliqué à l'observation bruitée X_c de longueur d modélisée par :

$$\mathsf{X}_c = \hat{\mathsf{X}} + \mathsf{N}_c,$$

où N_c est un vecteur aléatoire gaussien de matrice de covariance $\sigma^2\mathsf{I}_d$, et chaque séquence $\hat{\mathsf{X}} \in \mathcal{C}$ est supposée équiprobable, c'est à dire :

$$\forall \hat{\mathsf{x}} \in \mathcal{C}, \mathbb{P}(\hat{\mathsf{X}} = \hat{\mathsf{x}}) = \frac{1}{2^{nd}}.$$

La surface d'une hypersphère unitaire en dimension d sera notée $S_d = \frac{2\pi^{\frac{d}{2}}}{\Gamma(\frac{d}{2})}$, et son volume V_d.

TTCQ d'une source uniforme

La probabilité que X_s soit proche de sa représentation quantifiée $\hat{\mathsf{X}}$ est donnée par :

$$\mathbb{P}(\|\mathsf{X}_s - \hat{\mathsf{X}}\| < r) = \sum_{\hat{\mathsf{x}} \in \mathcal{C}} \mathbb{P}(\hat{\mathsf{X}} = \hat{\mathsf{x}}) \mathbb{P}(\|\mathsf{X}_s - \hat{\mathsf{X}}\| < r | \hat{\mathsf{X}} = \hat{\mathsf{x}})$$

Nous supposons que le bruit que quantification est indépendant de $\hat{\mathsf{X}}$, c'est à dire que toute cellule de Voronoï a même forme et même volume. Cette hypothèse est uniquement invalidée sur les bords de l'hypercube, ce qui correspond à un nombre négligeable de cellules à haut débit et en dimension élevée. En supposant que l'hypersphère de rayon r est entièrement incluse dans la cellule de quantification pour un r suffisamment petit, nous obtenons :

$$\mathbb{P}(\|\mathsf{X}_s - \hat{\mathsf{X}}\| < r | \hat{\mathsf{X}} = \hat{\mathsf{x}}) = \mathbb{P}(\|\mathsf{N}_s\| < r) = \int_{\|\mathsf{n}\| < r} \frac{2^{nd}}{a^d} d\mathsf{n}.$$

Comme la source est uniforme, les vecteurs quantifiés $\hat{\mathsf{X}}$ sont équiprobables, ce qui mène à :

$$\mathbb{P}(\|\mathsf{X}_s - \hat{\mathsf{X}}\| < r) = \sum_{\hat{x} \in \mathcal{C}} \frac{1}{2^{nd}} \int_{\|\mathsf{n}\| < r} \frac{2^{nd}}{a^d} d\mathsf{n} = \frac{2^{nd}}{a^d} V_d r^d.$$

En utilisant l'expression du volume d'une hypersphère unitaire en dimension d, dont l'expression est $V_d = \frac{S_d}{d} = \frac{\pi^{\frac{d}{2}}}{\Gamma(1+\frac{d}{2})}$, on obtient finalement :

$$\mathbb{P}(\|\mathsf{X}_s - \hat{\mathsf{X}}\| < r) = \frac{2^{nd}\pi^{\frac{d}{2}}}{a^d}\frac{r^d}{\Gamma(1+\frac{d}{2})}.$$

TTCM dans un canal AWGN

La probabilité que X_c soit proche du vecteur $\hat{\mathsf{X}}$ émis sur le canal est donnée par :

$$\mathbb{P}(\|\mathsf{X}_c - \hat{\mathsf{X}}\| < r) = \sum_{\hat{\mathsf{x}}\in\mathcal{C}} \mathbb{P}(\hat{\mathsf{X}} = \hat{\mathsf{x}})\mathbb{P}(\|\mathsf{X}_c - \hat{\mathsf{X}}\| < r|\hat{\mathsf{X}} = \hat{\mathsf{x}})$$

En utilisant l'expression du bruit gaussien, indépendant de $\hat{\mathsf{X}}$, nous avons :

$$\mathbb{P}(\|\mathsf{X}_c - \hat{\mathsf{X}}\| < r|\hat{\mathsf{X}} = \hat{\mathsf{x}}) = \mathbb{P}(\|\mathsf{N}_c\| < r) = \int\limits_{\|\mathsf{n}\|<r} \frac{1}{(2\pi\sigma^2)^{\frac{d}{2}}} e^{-\frac{\|\mathsf{n}\|^2}{2\sigma^2}} d\mathsf{n}.$$

Ainsi, en supposant les séquences modulées équiprobables, on obtient :

$$\mathbb{P}(\|\mathsf{X}_c - \hat{\mathsf{X}}\| < r) = \sum_{\hat{x}\in\mathcal{C}} \frac{1}{2^{nd}} \int\limits_{\|\mathsf{n}\|<r} \frac{1}{(2\pi\sigma^2)^{\frac{d}{2}}} e^{-\frac{\|\mathsf{n}\|^2}{2\sigma^2}} d\mathsf{n} = \frac{1}{(2\pi\sigma^2)^{\frac{d}{2}}} \int\limits_{\|\mathsf{n}\|<r} e^{-\frac{\|\mathsf{n}\|^2}{2\sigma^2}} d\mathsf{n}.$$

En intégrant sur la surface d'une hypersphère de rayon variable x, et en utilisant le changement de variable $u = \frac{x^2}{2\sigma^2}$ nous obtenons :

$$\mathbb{P}(\|\mathsf{X}_c - \hat{\mathsf{X}}\| < r) = \frac{1}{(2\pi\sigma^2)^{\frac{d}{2}}} \in t_0^r S_d x^{d-1} e^{-\frac{x^2}{2\sigma^2}} dx = \frac{S_d}{2\pi^{\frac{d}{2}}} \int_0^{\frac{r^2}{2\sigma^2}} u^{\frac{d}{2}-1} e^{-u} du$$

qui peut s'exprimer à l'aide de la fonction gamma incomplète inférieure de la manière suivante :

$$\mathbb{P}(\|\mathsf{X}_c - \hat{\mathsf{X}}\| < r) = \frac{S_d\gamma(\frac{d}{2}, \frac{r^2}{2\sigma^2})}{2\pi^{\frac{d}{2}}} = \frac{\gamma(\frac{d}{2}, \frac{r^2}{2\sigma^2})}{\Gamma(\frac{d}{2})}.$$

La fonction gamma incomplète inférieure $\gamma(a, z)$ se développe au voisinage de $z = 0$ en $\gamma(a, z) \approx_{z=0} \frac{1}{a}z^a + o(z^{a+1})$. Ainsi, pour un petit rayon r,

$$\mathbb{P}(\|\mathsf{X}_c - \hat{\mathsf{X}}\| < r) \approx \frac{2}{d}\left(\frac{r^2}{2\sigma^2}\right)^{\frac{d}{2}} \frac{1}{\Gamma(\frac{d}{2})} = \frac{1}{(2\sigma^2)^{\frac{d}{2}}} \frac{r^d}{\frac{d}{2}\Gamma(\frac{d}{2})}.$$

En utilisant la propriété $x\Gamma(x) = \Gamma(x+1)$ on obtient finalement :

$$\mathbb{P}(\|\mathsf{X}_c - \hat{\mathsf{X}}\| < r) = \frac{1}{(2\sigma^2)^{\frac{d}{2}}} \frac{r^d}{\Gamma(1+\frac{d}{2})}.$$

Comparaison des deux systèmes

Comparons à présent la TTCQ et la TTCM pour un même rapport signal à bruit. La puissance du bruit de quantification scalaire sur n bits d'une source uniformément distribuée sur $[0, a]$ est donnée par :

$$E[\mathsf{N}^2] = \int_{-\frac{1}{2}\frac{a}{2^n}}^{\frac{1}{2}\frac{a}{2^n}} \frac{2^n}{a} x^2 dx = \frac{1}{2^{2n}} \frac{a^2}{12}.$$

Pour la TTCM par modulation d'amplitude sur 2^n niveaux sur un canal AWGN, l'énergie du bruit est donnée par :

$$E[\mathsf{N}^2] = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}\sigma} x^2 e^{-\frac{x^2}{2\sigma^2}} dx = \sigma^2.$$

Comme les deux systèmes partagent le même dictionnaire $\mathcal{C}$, l'energie du signal $\hat{X}$ est identique dans les deux cas. Nous supposons que ce dictionnaire a une distance minimale euclidienne plus importante que le dictionnaire correspondant à une quantification scalaire sur le même nombre de bits (sinon le quantificateur TTCQ n'a aucun intérêt). Alors la puissance du bruit de quantification σ_s^2 est bornée par :

$$\sigma_s^2 < \frac{a^2}{12 \cdot 2^{2n}}.$$

Pour un même rapport signal à bruit, la variance σ^2 du bruit pour la TTCM doit alors aussi vérifier cette contrainte :

$$\sigma^2 < \frac{a^2}{12 \cdot 2^{2n}}.$$

En combinant (5.10) et (5.10) nous avons :

$$\mathbb{P}(\|\mathsf{X}_s - \hat{\mathsf{X}}\| < r) = \frac{2^{nd} \pi^{\frac{d}{2}}}{a^d} (2\sigma^2)^{\frac{d}{2}} \mathbb{P}(\|\mathsf{X}_c - \hat{\mathsf{X}}\| < r).$$

Alors,

$$\mathbb{P}(\|\mathsf{X}_s - \hat{\mathsf{X}}\| < r) < \frac{2^{nd} \pi^{\frac{d}{2}}}{a^d} (2 \frac{a^2}{12 \cdot 2^{2n}})^{\frac{d}{2}} \mathbb{P}(\|\mathsf{X}_c - \hat{\mathsf{X}}\| < r)$$

$$\mathbb{P}(\|\mathsf{X}_s - \hat{\mathsf{X}}\| < r) < (\frac{\pi}{6})^{\frac{d}{2}} \mathbb{P}(\|\mathsf{X}_c - \hat{\mathsf{X}}\| < r).$$

La probabilité d'observer X au voisinage d'un centroïde dans la TTCQ d'une source uniforme est plus faible que la probabilité d'observer une séquence bruitée proche de la séquence modulée par TTCM sur canal AWGN, au même rapport signal à bruit. Ainsi le même algorithme itératif appliqué dans les deux cas à bien moins de chance de

converger pour la TTCQ du fait d'une bien moins bonne initialisation. En supposant que l'ensemble des vecteurs pour lesquels l'algorithme converge vers un point fixe réside dans une sphère de convergence de rayon r_C, on obtient une nombre exponentiellement plus faible de solutions correctes pour la TTCQ que pour la TTCM à mesure que la longueur de la séquence augmente.

Glossaire

3D-ESCOT : *Three-dimensional embedded subband coding with optimized truncation (codage de sous-bandes 3D avec troncature optimisée)*, extension du codeur EBCOT pour le codage de sous-bandes d'ondelettes vidéo 2D+t.

AWGN : *Additive White Gaussian Noise (bruit blanc additif gaussien)*, se dit d'un canal de transmission modélisé par l'ajout d'un bruit gaussien indépendant au signal modulé.

bpp : *Bits par pixels.*

CABAC : *Context Adaptive Binary Arithmetic Coder (codeur arithmétique binaire contextuel adaptatif)*, technique de codage arithmétique adaptatif utilisée dans la norme H.264.

DWT : *Discrete Wavelet Transform (transformée en ondelettes discrètes).*

EBCOT : *Embedded Block Coder with Optimal Truncation (codeur emboîté par blocs avec troncature optimale)*, codeur de sous-bandes par blocs avec optimisation débit-distorsion a posteriori utilisé dans la norme JPEG2000.

EQ : *Estimation-Quantization (estimation/quantification)*, codeur de sous-bandes non progressif basé sur un modèle adaptatif de gaussienne généralisées.

EM : *Expectation Maximization (espérance/maximisation)*, algorithme itératif d'estimation reposant sur deux étapes complémentaires de calcul d'espérance et maximisation de la vraisemblance.

EZBC : *Embedded Zero Block Coder (codeur emboîté par blocs de zéros)*, codeur de sous-bandes par blocs.

EZW : *Embedded coding with Zero tree of Wavelet coefficients (codage emboîté par arbres de coefficients d'ondelettes nuls)*, premier codeur de sous-bandes progressif.

H.264 : norme de compression vidéo issue des organismes de normalisation ITU et

ISO.

HMF : *Hidden Markov Field (champ de Markov caché).*

JPEG2000 : norme de compression d'image par ondelettes issue de l'organisme de normalisation ISO et utilisant le codeur EBCOT.

LDPC : *Low Density Parity Check (faible densité de vérification de parité)*, se dit de codes linéaires dont la matrice de parité est creuse et dont le décodage s'effectue itérativement par inférence bayésienne sur le graphe correspondant.

MAP : *Maximum A Posteriori.*

MC-EZBC : *Motion Compensated EZBC (EZBC compensé en mouvement)*, codage vidéo par ondelettes 2D+t basé sur le codeur EZBC et la transformée temporelle MCTF.

MCTF : *Motion Compensated Temporal Filtering (filtrage temporel compensé en mouvement)*, technique de filtrage par ondelettes le long des trajectoires de mouvement en vidéo.

MCWC : *Motion Compensated Wavelet Coder (codeur par ondelettes compensées en mouvement)*, codage vidéo par ondelettes 2D+t basé sur le codeur 3D-ESCOT et la norme H.264.

MPEG : *Motion Picture Expert Group (groupe d'experts en vidéo)*, groupe de travail de l'organisme de normalisation ISO visant à la création de standards vidéo.

MSE : *Mean Square Error (erreur quadratique moyenne).*

codeur MQ : codeur quasi-arithmétique adaptatif par automate (sans multiplication).

POCS : *Projections on Convex Sets (projections sur ensembles convexes)*, application du théorème du point fixe à la suite des projections successive sur des ensembles convexes.

PSNR : *Peak Signal to Noise Ratio (rapport signal à bruit crète à crète)*, mesure objective de qualité d'un signal borné.

RD : *Rate-Distortion (débit-distorsion).*

SFQ : *Space Frequency Quantization (quantification temps-fréquence)*, technique de codage en sous-bandes non progressive.

SNR : *Signal to Noise Ratio (rapport signal à bruit).*

SOTCQ : *Soft Output Trellis-Coded Quantization (quantification codée par trellis à sortie souple)*, extension de la TCQ fournissant une information sur les distorsions relatives entre différents choix de centroïdes.

SOVA : *Soft Output Viterbi Algorithm (algorithme de Viterbi à sortie souple)*, extension de l'algorithme de Viterbi fournissant une information sur les métriques relatives entre différents chemins du treillis.

SPIHT : *Set Partitioning In Hierarchical Trees (partitionnement d'ensembles en arbres hiérarchiques)*, codeur en sous-bandes progressif par arbres de zéros.

TCQ : *Trellis-Coded Quantization (quantification codée par trellis)*, technique de quantification reposant sur des codes correcteurs d'erreur convolutifs.

TCM : *Trellis-Coded Modulation (modulation codée par trellis)*, technique de modulation reposant sur des codes correcteurs d'erreur convolutifs.

TTCM : *Turbo TCM*, technique de modulation TCM utilisant des codes turbo au lieu de codes convolutifs.

TTCQ : *Turbo TCQ*, technique de quantification TCQ utilisant des codes turbo au lieu de codes convolutifs.

VQ : *Vector Quantization (quantification vectorielle)*.

Publications

Journaux Internationaux

[1] V. Chappelier, C. Guillemot
"Oriented Wavelet Transform for Image Compression and Denoising,"
soumis à *IEEE Transactions on Image Processing.*

Conférences Internationales

[2] V. Chappelier, C. Guillemot
"Oriented Wavelet Transform on a Quincunx Pyramid for Image Compression,"
IEEE International Conference on Image Processing, Sept. 2005.

[3] V. Chappelier, C. Guillemot, S. Marinkovic,
"Image coding with iterated contourlet and wavelet transforms,"
IEEE International Conference on Image Processing, Oct. 2004.

[4] V. Chappelier, C. Guillemot, S. Marinkovic,
"Turbo trellis coded quantization,"
International Symposium on Turbo Codes, Sept. 2003, pp. 51–54.

Conférences Nationales

[5] V. Chappelier, C. Guillemot, S. Marinkovic,
"Codage d'images par ondelettes unidimensionnelles orientées,"
CORESA, 2004, pp. 117–120.

Bibliographie

[1] J. H. Conway and N. J. Sloane, *Sphere Packings, Lattices, and Groups, 2nd ed. New York*, Springer-Verlag, 1993.

[2] P. P. Vaidyanathan, *Multirate Systems and Filter Banks*, Prentice Hall P T R, 1993.

[3] L. Soderberg, "Swedish accent," *Playboy*, pp. 134–141, Nov. 1972.

[4] M. N. Do and M. Vetterli, "Framing pyramids," in *IEEE Transactions on Signal Processing*, Sept. 2003.

[5] J. C. Feauveau, "Analyse multirésolution avec un facteur de résolution $\sqrt{2}$," in *Journal de Traitement du Signal*, 1990, vol. 7 (2), pp. 117–128.

[6] J. H. McClellan, "The design of two-dimensional digital filters by transformation," in *7th Annual Princeton Conference on Information Sciences and Systems*, Mar. 1973, pp. 247–251.

[7] S. M. Phoong, C. W. Kim, P. P. Vaidyanathan, and R. Ansari, "A new class of two-channel biorthogonal filter banks and wavelet bases," in *IEEE Transactions on Signal Processing*, Mar. 1995, vol. 43(3), pp. 649–665.

[8] A. Gouze, M. Antonini, and M. Barlaud, "Quincunx lifting scheme for lossy image coding," in *IEEE International Conference on Image Processing*, Sept. 2000, vol. 1, pp. 665–668.

[9] J. Kovačević and M. Vetterli, "Nonseparable multidimensional perfect reconstruction filter banks and wavelet bases for $\mathbb{R}^n$," *IEEE Transactions on Information Theory*, vol. 38, no. 2, pp. 533–555, Mar. 1992.

[10] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting steps," in *Journal of Fourier Analysis and Applications*, 1998, vol. 4(3), pp. 247–269.

[11] Euclide d'Alexandrie, *Éléments*, vol. 7, environ 300 avant J.C.

[12] M. Antonini, M. Barlaud, P. Mathieu, , and I. Daubechies, "Image coding using wavelet transform," in *IEEE Transactions on Image Processing*, Apr. 1992, vol. 1, pp. 205–220.

[13] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, John Wiley & Sons, Inc., New York, N.Y., 1991.

[14] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423 et 623–656, July and October 1948.

[15] R. C. Pasco, *Source Coding Algorithms for Fast Data Compression*, Ph.D. thesis, Stanford University, CA, May 1976.

[16] J. J. Rissanen, "Generalized kraft inequality and arithmetic coding," Tech. Rep., 1976.

[17] N. Abramson, *Information theory and coding*, McGraw-Hill, New York, 1963.

[18] P. G. Howard and J. S. Vitter, "Practical implementations of arithmetic coding," Tech. Rep. CS-91-45, 1991.

[19] D. Taubman, "High performance scalable image compression with EBCOT," *IEEE Transactions on Image Processing*, vol. 9(7), pp. 1158–1170, 2000.

[20] D. Marpe, H. Schwarz, and T. Wiegand, "Context-based adaptive binary arithmetic coding in the h.264/avc video compression standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13(7), pp. 620–636, July 2003.

[21] R. E. Blahut, "Computation of channel capacity and rate-distortion function," in *IEEE Transactions on Information Theory*, July 1972, vol. 18(4), pp. 460–473.

[22] R. M. Gray and D. L. Neuhoff, "Quantization," *IEEE Transactions on Information Theory*, vol. 44, pp. 2325–2384, Oct. 1998.

[23] J. Max, "Quantizing for minimum distortion," *IRE Transactions on Information Theory*, vol. 6(1), pp. 7–12, Mar. 1960.

[24] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28(2), pp. 129–137, Mar. 1982.

[25] P. A. Chou, T. Lookabaugh, and R. M. Gray, "Entropy constrained vector quantization," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, Jan. 1989.

[26] J. O. Coleman, "Coset decomposition in lattices yields sample-block number systems," in *IEEE International Symposium on Circuits and Systems*, May 2002.

[27] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications*, vol. 28, no. 1, Jan. 1980.

[28] A. Gerscho and R. M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publisher, 1992.

[29] J. H. Conway and N. J. A. Sloane, "A lower bound on the average error of vector quantizers," *IEEE Transactions on Information Theory*, vol. 31, no. 1, pp. 106–109, 1985.

[30] M. Vetterli, "Wavelets, approximation and compression," in *IEEE Signal Processing Magazine*, Sept. 2001, pp. 59–73.

[31] F. Friedrich H. Führ, L. Demaret, "Beyond wavelets : New image representation paradigms," in *Survey article, preprint version*, 2005.

[32] P. L. Dragotti and M. Vetterli, "Wavelet footprints : Theory, algorithms and applications," *IEEE Transactions on Signal Processing*, vol. 51(5), pp. 1306–1323, May 2003.

[33] J. Radon, "Über die bestimmung von funktionen durch ihre integralwerte längs gewisser mannigfaltigkeiten," in *Berichte Saechsische Akademie der Wissenschaften, Leipzig. Math. Nat*, 1917, vol. 69, pp. 262–277.

[34] P. V. C. Hough, "Machine analysis of bubble chamber pictures," in *International Conference on High Energy Accelerators and Instrumentation*, 1959.

[35] R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," in *Communications of the ACM*, Jan. 1972, vol. 15, pp. 11–15.

[36] F. Matús and J. Flusser, "Image representation via a finite radon transform," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15(10), pp. 996–1006, Oct. 1993.

[37] M. L. Brady and W. Yong, "Fast parallel discrete approximation algorithms for the radon transform," in *4th Annual ACM Symposium on Parallel Algorithms and Architectures*, 1992, pp. 91–99.

[38] E. J. Candes, *Ridgelets : Theory and Applications*, Ph.D. thesis, Department of Statistics, Stanford University, 1998.

[39] M. N. Do and M. Vetterli, "Orthonormal finite ridgelet transform for image compression," in *IEEE International Conference on Image Processing*, 2000, vol. 2, pp. 367–370.

[40] D. L. Donoho, "Orthonormal ridgelets and linear singularities," Tech. Rep., Stanford University, 1998.

[41] E. J. Candes and D. L. Donoho, *Curvelets – a surprisingly effective nonadaptive representation for objects with edges*, pp. 1–10, Vanderbilt University Press, Nashville, TN, 1999.

[42] E. J. Candes and D. L. Donoho, "New tight frames of curvelets and optimal representations of objects with smooth singularities," Tech. Rep., Stanford University, 2002.

[43] J. L. Starck, E. Candes, and D. L. Donoho, "The curvelet transform for image denoising," *IEEE Transactions on Image Processing*, vol. 11(6), pp. 670–684, 2002.

[44] M. Do and M. Vetterli, "Pyramidal directional filter banks and curvelets," in *IEEE International Conference on Image Processing*, 2001.

[45] M. N. Do and M. Vetterli, "The contourlet transform : An efficient directional multiresolution image representation," *IEEE Transactions on Image Processing*, Oct. 2003.

[46] H. Bamberger and M. J. T. Smith, "A filterbank for the directional decomposition of images : Theory and design," *IEEE Transactions on Signal Processing*, vol. 40(4), pp. 882–893, Apr. 1992.

[47] P. J. Burt and E. H. Adelson, "The laplacian pyramid as a compact image code," *IEEE Transactions on Communications*, vol. 31(4), pp. 532–540, 1983.

[48] R. Eslami and H. Radha, "Wavelet-based contourlet coding using an SPIHT-like algorithm," in *IEEE International Conference on Image Processing*, Oct. 2004.

[49] A. Said and W. A. Pearlman, "A new, fast and efficient image codec based on set partitioning in hierarchical trees," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, pp. 243–250, 1996.

[50] Y. Lu and M. N. Do, "Crisp-contourlets : a critically sampled directional multiresolution image representation," in *SPIE Conference on Wavelet Applications in Signal and Image Processing*, Aug. 2003.

[51] N. G. Kingsbury, "The dual-tree complex wavelet transform : a new efficient tool for image restoration and enhancement," in *European Signal Processing Conference*, Sept. 1998, pp. 319–322.

[52] N. G. Kingsbury, "Shift invariant properties of the dual-tree complex wavelet transform," in *IEEE Conference on Acoustics, Speech and Signal Processing*, 1999.

[53] P. De Rivaz, *Complex Wavelet Based Image Analysis and Synthesis*, Ph.D. thesis, University of Cambridge, UK, 2000.

[54] F. Fernandes, *Directional, Shift-Insensitive, Complex Wavelet Transforms with Controllable Redundancy*, Ph.D. thesis, Rice, Houston, TX, August 2001.

[55] T. N. T. Goodman and S. L. Lee, "Wavelets of multiplicity r," *Transactions of the American Mathematical Society*, vol. 342(1), pp. 307–324, Mar. 1994.

[56] N. Kingsbury and T. Reeves, "Iterative image coding with overcomplete complex wavelet transforms," in *SPIE Visual Communications and Image Processing*, July 2003, vol. 5150, pp. 1253–1264.

[57] A. B. Watson, "The cortex transform : rapid computation of simulated neural images," *Computer Vision, Graphics, and Image Processing*, vol. 39(3), pp. 311–327, Sept. 1987.

[58] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *Journal of Physiology (London)*, vol. 160, pp. 106–154, 1962.

[59] D. H. Hubel and T. N. Wiesel, "Sequence regularity and geometry of orientation columns in the monkey striate cortex," *Journal of Comparative Neurology*, vol. 158, pp. 267–293, 1974.

[60] S. Daly, "The visible differences predictor : An algorithm for the assessment of image fidelity," in *A. B. Watson, Digital Image and Human Vision*. 1993, pp. 179–206, MIT Press.

[61] A. B. Watson, "Efficiency of a model human image code," *Journal of the Optical Society of America A*, vol. 4, pp. 2401–2417, 1987.

[62] J. Pinilla-Dutoit and S. I. Woolley, "Robust perceptual coding of overcomplete frame expansions," in *SPIE Human Vision and Electronic Imaging*, 2001, vol. 4299, pp. 143–149.

[63] E. P. Simoncelli and E. H. Adelson, "Noise removal via bayesian wavelet coring," in *IEEE International Conference on Image Processing*, Sept. 1996.

[64] A. Karasaridis and E. P. Simoncelli, "Filter design technique for steerable pyramid image transforms," in *IEEE Conference on Acoustics, Speech and Signal Processing*, May 1996.

[65] E. P. Simoncelli and W. T. Freeman, "The steerable pyramid : A flexible architecture for multi-scale derivative computation," in *IEEE International Conference on Image Processing*, Nov. 1995.

[66] E. P. Simoncelli, W. T. Freeman, E. H. Adelson, and D. J. Heeger, "Shiftable multiscale transforms," *IEEE Transactions on Information Theory : Special Issue on Wavelets*, vol. 38(2), pp. 587–607, Mar. 1992.

[67] W. T. Freeman and E. H. Adelson, "The design and use of steerable filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13(9), pp. 891–906, Sept. 1991.

[68] B. Beferull-Lozano and A. Ortega, "Coding techniques for oversampled steerable transforms," in *International Asilomar Conference on Signals, Systems and Computers*, Oct. 1999.

[69] S. G. Mallat and Z. Zhang, "Matching pursuit with time-frequency dictionaries," in *IEEE Transactions on Signal Processing*, Dec. 1993, vol. 41, pp. 3397–3415.

[70] M. Sabin and R. Gray, "Product code vector quantizers for speech waveform coding," in *Globecom*, Dec. 1982, pp. 1087–1091.

[71] M. J. Sabin and R. M. Gray, "Product code vector quantizers for waveform and voice coding," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 32, pp. 474–488, June 1984.

[72] R. R. Coifman, Y. Meyer, and M. V. Wickerhauser, "Size properties of wavelet-packets," in *Wavelets and their applications*, pp. 453–470. Jones and Bartlett, Boston, MA, 1992.

[73] R. R. Coifman, Y. Meyer, and M. V. Wickerhauser, "Adapted wave form analysis, wavelet-packets and applications," in *International Conference on Industrial and Applied Mathematics*, pp. 41–50. 1991.

[74] K. Ramchandran and M. Vetterli, "Best wavelet packet bases in a rate-distorsion sense," *IEEE Transactions on Image Processing*, vol. 2, pp. 160–175, Apr. 1993.

[75] F. G. Meyer and R. R. Coifman, "Brushlets : a tool for directional image analysis and image compression," in *Applied and Computational Harmonic Analysis*, 1997, vol. 4, pp. 147–187.

[76] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Transactions on Signal Processing : Special Issue Wavelets and Signal Processing*, vol. 41, pp. 3445–3463, 1993.

[77] D. L. Donoho and X. Huo, "Beamlet pyramids : a new form of multiresolution analysis, suited for extracting lines, curves and objects from very noisy image data," in *SPIE Conference on Wavelet Applications in Signal and Image Processing*, 2000, vol. 4119, pp. 434–444.

[78] D. L. Donoho, "Wedgelets : Nearly minimax estimation of edges," in *Annals of Statistics*, 1999, vol. 27(3), pp. 859–897.

[79] M. Wakin, J. Romberg, H. Choi, and R. Baraniuk, "Rate-distortion optimized image compression using wedgelets," in *IEEE International Conference on Image Processing*, Sept. 2002.

[80] Z. Xiong, K. Ramchandran, and M. T. Orchard, "Space-frequency quantization for wavelet image coding," *IEEE Transactions on Image Processing*, vol. 6(5), pp. 677–693, 1997.

[81] V. Chandrasekaran, M. Wakin, D. Baron, and R. Baraniuk, "Surflets : A sparse representation for multidimensional functions containing smooth discontinuities," in *IEEE International Symposium on Information Theory*, Chicago, IL, June 2004.

[82] R. M. Willett and R. D. Nowak, "Platelets : A multiscale approach for recovering edges and surfaces in photon-limited medical imaging," Tech. Rep. TREE0105, Rice University, 2002.

[83] E. Pennec and S. Mallat, "Image compression with geometrical wavelets," in *IEEE International Conference on Image Processing*, 2000, vol. 1, pp. 661–664.

[84] E. Le Pennec and S. Mallat, "Sparse geometric image representation with bandelets," in *submitted to IEEE Transactions on Image Processing*, 2003.

[85] C. Bernard and E. Le Pennec, "Adaptation of regular grid filterings to irregular grids," Tech. Rep., École polytechnique, Centre de Mathématiques Appliquées, 2002.

[86] G. Peyré and S. Mallat, "Surface compression with geometric bandelets," in *SIGGRAPH*, Aug. 2005.

[87] M. Vetterli V. Velisavljevic, B. Beferull-Lozano and P.L. Dragotti, "Directionlets : Anisotropic multi-directional representation with separable filtering," *submitted to IEEE Transactions on Image Processing*, Dec. 2004.

[88] V. Velisavljevic, B. Beferull-Lozano, M. Vetterli, and P. L. Dragotti, "Approximation power of directionlets," in *IEEE International Conference on Image Processing*, Sept. 2005.

[89] S. M. LoPresto, K. Ramchandran, and M. T. Orchard, "Image coding based on mixture modeling of wavelet coefficients and a fast estimation-quantization framework," in *IEEE Data Compression Conference*, 1997, pp. 221–230.

[90] P. Sriram and M. W. Marcellin, "Wavelet coding of images using trellis coded quantization," in *SPIE Conference on Visual Information Processing*, Apr. 1992, pp. 238–247.

[91] Y. H. Kim and J. M. Modestino, "Adaptive entropy coded subband coding of images," *IEEE Transactions on Image Processing*, vol. 1, pp. 31–48, Jan. 1992.

[92] A. Islam and W. A. Pearlman, "An embedded and efficient low-complexity hierarchical image coder," in *SPIE Visual Communications and Image Processing*, Jan. 1999, vol. 3653, pp. 294–305.

[93] A. Said and W. A. Pearlman, "Low-complexity waveform coding via alphabet and sample-set partitioning," in *SPIE Visual Communications and Image Processing*, Feb. 1997, vol. 3024, pp. 25–37.

[94] J. Andrew, "A simple and efficient hierarchical image coder," in *IEEE International Conference on Image Processing*, 1997, vol. 3(3), p. 658.

[95] W. B. Pennebaker, J. L. Mitchell, G. G. Langdon, and R. B. Arps, "An overview of the basic principles of the q-coder adaptive binary arithmetic coder," *IBM Journal of Research and Development*, vol. 32(6), pp. 717–726, Nov. 1988.

[96] "Proposal of the arithmetic coder for JPEG2000," in *ISO/IEC JTC1/SC29/WG1 N762*, Mar. 1998.

[97] J. Xu, Z. Xiong, S. Li, and Y. Q. Zhang, "Three-dimensional embedded subband coding with optimized truncation (3D ESCOT)," in *Applied and Computational Harmonic Analysis*, 2001, vol. 10, pp. 290–315.

[98] S. Hsiang and J. W. Woods, "Embedded image coding using zeroblocks of subband/wavelet coefficients and context modeling," in *MPEG-4 Workshop and Exhibition at ISCAS 2000*, May 2000.

[99] S. Hsiang and J. W. Woods, "Embedded video coding using invertible motion compensated 3-d subband/wavelet filter bank," *Signal Processing : Image Communications*, vol. 16, pp. 705–724, May 2001.

[100] J. Villasenor, B. Belzer, and J. Liao, "Wavelet filter evaluation for image compression," *IEEE Transactions on Image Processing*, vol. 2, pp. 1053–1060, Aug. 1995.

[101] G. Bjøntegaard and K. O. Lillevo, "H.263 anchors - technical description," Nov. 1995.

[102] B. Usevitch, "Optimal bit allocation for biorthogonal wavelet coding," in *IEEE Data Compression Conference*, 1996, pp. 387–395.

[103] Y. Yang and N. P. Galatsanos, "Removal of compression artifact using projections onto convex sets and line process modelling," in *IEEE Transactions on Image Processing*, Oct. 1997, vol. 6, pp. 1345–1357.

[104] S. Li W. Ding, F. Wu, "Lifting-based wavelet transform with directionally spatial prediction," in *Picture Coding Symposium*, Dec. 2004.

[105] J. Liu and P. Moulin, "Information-theoretic analysis of interscale and intrascale dependencies between image wavelet coefficients," *IEEE Transactions on Image Processing*, vol. 10, no. 10, pp. 647–1658, Nov. 2001.

[106] P. Clifford, "Markov random fields in statistics," in *Disorder in physical systems, A volume in honour of J. M. Hammersley*, G. R. Grimmett and D. J. A. Welsh, Eds. Clarendon Press, Oxford, 1990.

[107] G. Fan and X.-G. Xia, "Image denoising using local contextual hidden markov model in the wavelet domain," *IEEE Signal Processing Letters*, vol. 8, no. 5, pp. 125–128, May 2001.

[108] J. Portilla, V. Strela, M. Wainwright, and E. P. Simoncelli, "Image denoising using scale mixtures of gaussians in the wavelet domain," *IEEE Transactions on Image Processing*, vol. 12(11), pp. 1338–1351, Nov. 2003.

[109] E. Le Pennec, *Bandelettes et représentation géométrique des images*, Ph.D. thesis, Ecole Polytechnique, Dec. 2002.

[110] J. H. Kasner, M. W. Marcellin, and B. R. Hunt, "Universal trellis coded quantization," *submitted to IEEE Transactions on Image Processing*, 1995.

[111] R. L. Joshi, V. J. Crump, and T. R. Fischer, "Image subband coding using arithmetic and trellis coded quantization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, pp. 515–523, Dec. 1995.

[112] A. Bilgin, P. J. Sementilli, and M. W. Marcellin, "Progressive image coding using trellis coded quantization," *IEEE Transactions on Image Processing*, vol. 8(11), pp. 1638–1643, 1999.

[113] R. L. Joshi, H. Jafarkhani, J. H. Kasner, T. R. Fischer, N. Farvardin, M. W. Marcellin, and R. H. Bamberger, "Comparison of different methods of classification in subband coding of images," *IEEE Transactions on Image Processing*, vol. 6, pp. 1473–1487, 1997.

[114] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding : Turbo codes," in *IEEE International Conference on Communications*, May 1993, pp. 1064–1070.

[115] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, vol. IT-20, pp. 284–287, Mar. 1974.

[116] J. Hagenauer and L. Papke, "Decoding "turbo"-codes with the soft output viterbi algorithm," in *IEEE International Symposium on Information Theory*, June 1994, p. 164.

[117] G. Ungerboeck, "Channel coding with multilevel/phase signals," *IEEE Transactions on Information Theory*, vol. IT-28, no. 1, pp. 55–67, Jan. 1982.

[118] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Parallel concatenated trellis coded modulation," in *IEEE International Conference on Communications*, 1996, vol. 2, pp. 974–978.

[119] P. Robertson and T. Wörz, "Bandwidth-efficient turbo trellis-coded modulation using punctured component codes," *IEEE Journal on Selected Areas on Communication*, vol. 16, no. 2, pp. 206–218, Feb. 1998.

[120] M. W. Marcellin and T. R. Fisher, "Trellis-coded quantization of memoryless and gauss-markov sources," *IEEE Transactions on Communications*, vol. 38, pp. 82–93, Jan. 1990.

[121] J. Garcia-Frias and J. Zhao, "Compression of binary memoryless sources using punctured turbo codes," in *IEEE Communications Letters*, Sept. 2002, vol. 6, pp. 394–396.

[122] J. Garcia-Frias and Y. Zhoa, "Compression of correlated binary sources using turbo codes," in *IEEE Communications Letters*, Oct. 2001, pp. 417–419.

[123] J. Bajcsy and P. Mitran, "Coding for the Slepian-Wolf problem with turbo codes," in *Globecom*, Dec. 2001.

[124] A. Aaron and B. Girod, "Compression with side information using turbo codes," in *IEEE Data Compression Conference*, Apr. 2002.

[125] G. Ungerboeck, "Trellis-coded modulation wth redundant signal sets, parts i and ii," *IEEE Communications Magazine*, vol. 25, pp. 5–21, Feb. 1987.

[126] A. Viterbi, "Error bounds for convolution codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, pp. 260–269, 1967.

[127] J. Pearl, *Probabilistic Reasoning in Intelligent Systems, Networks of Plausible Inference*, Morgan Kaufmann Publishers, Inc., 1988.

[128] R. G. Gallager, "Low-density parity-check codes," *IRE Transactions on Information Theory*, vol. 8, pp. 21–28, Jan. 1962.

[129] D. J. C. MacKay and R. M. Neal, "Near shannon limit performance of low density parity check codes," *IEE Electronics Letters*, vol. 32, no. 18, pp. 1645–1655, Aug. 1996.

[130] K. P. Murphy, Y. Weiss, and M. Jordan., "Loopy belief propagation for approximate inference : an empirical study," in *Uncertainty in AI*, 1999, vol. 9, pp. 467–475.

[131] M. Kearns and L. Ortiz, "Nash propagation for loopy graphical games," in *International Conference on Neural Information Processing Systems*, 2002.

[132] T. R. Fischer, M. W. Marcellin, and M. Wang, "Trellis-coded vector quantization," *IEEE Transactions on Information Theory*, vol. IT-37, pp. 1551–1566, Nov. 1991.

[133] T. Eriksson, M. Novak, and J. B. Anderson, "MAP criterion trellis source coding for short data sequences," in *IEEE Data Compression Conference*, Mar. 2003, pp. 43–52.

[134] R. J. Van Der Sleuten and J. H. Weber, "Construction and evaluation of trellis-coded quantizers for memoryless sources," in *IEEE Transactions on Information Theory*, May 1995, vol. 41(3), pp. 853–859.

[135] M. H. M. Costa, "Writing on dirty paper," *IEEE Transactions on Information Theory*, vol. 29(3), pp. 439–441, May 1983.

[136] D. Slepian and J. K. Wolf, "Noiseless coding of correlated information sources," *IEEE Transactions on Information Theory*, vol. 19, pp. 471–480, July 1973.

[137] A. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Transactions on Information Theory*, vol. 22, pp. 1–10, Jan. 1976.

[138] Z. Xiong, V. Stanković, S. Cheng, A. Liveris, and Y. Sun, "Source-channel coding for algebraic multiterminal binning," in *IEEE Information Theory Workshop*, Oct. 2004.

[139] G. Le Gelvouit, "Trellis-coded quantization for public-key steganography," in *IEEE Conference on Acoustics, Speech and Signal Processing*, Mar. 2005.

Table des figures

Résumé

Dans cette thèse, nous nous intéressons à la conception de transformées orientées pour la compression d'images. La transformée en contourlettes redondante est adaptée à la compression en la combinant avec les ondelettes séparables. Une nouvelle transformée adaptative est ensuite conçue, où l'information géométrique de l'image est décrite de manière explicite. Cette transformée en ondelettes orientées repose sur une décomposition multiéchelle quinconce où les pas lifting d'une ondelette sont orientés localement suivant une carte d'orientation. Ces transformées sont couplées aux codeurs de sous-bande EZBC et EBCOT pour l'évaluation des performances de bout en bout. Dans le cadre de la compression avec pertes, nous étudions également l'utilisation de turbo codes pour la quantification codée par treillis (TCQ). En utilisant la dualité source-canal, nous proposons une structure de quantification turbo TCQ et avons évaluons ses performances sur des sources sans mémoire.

Abstract

In this thesis, we focus on the design of oriented transforms for image compression. The redundant contourlet transform is adapted to compression by combining it with separable wavelets. A new adaptive transform is also proposed, in which the geometry of the image is described explicitly. This oriented wavelet transform is based on a multiscale quincunx decomposition where the lifting steps of a wavelet are oriented locally according to an orientation map. These transforms are combined with the EZBC and EBCOT subband coders for performance evaluation. In the context of lossy compression, we study the use of turbo codes for trellis-coded quantization (TCQ). Using the source-channel duality, we propose a turbo TCQ structure and evaluate its performance on memoryless sources.